NASA CR 70708

INTERIM REPORT NO. 1

HUMAN PERFORMANCE CONTROL MONITORING SYSTEM

Contract No. NASW 1085

21 Dec. 1964 through 30 Sept. 1965

TABLE OF CONTENTS

LIST OF ILLUSTRATIONS

1.0 INTRODUCTION

1.1 Summary

This Interim Report describes the work performed under Contract No. NASW-1085 in the period 22 December 1964 through 30 September 1965.

Theoretical studies include the development of a mathematical model of a performance control and monitoring system, development of techniques for prediction and estimation, and a description of decision methods and criteria.

Applications of trainable logic have been developed in the areas of computation and control. In the area of computation, Monte Carlo type methods and the solution of simultaneous equations have been investigated using statistical switch techniques.

In addition, three application problems are presented. The selection of one of these problems by NASA will be the basis for the simulation in the second phase of the program.

1.2 Historical Background

The study of trainable networks started with the investigation of neuron-like electronic elements, especially the ARTRON (artificial neuron). The objective of that study was to determine if these networks coulc be employed as engineering tools. This work included study of:

a. methods of information handling

b. methods of network interconnection

c. methods of network construction

d. methods of efficient use of redundant circuits

e. the theory of the organization process

f. the training criteria.

The ARTRON is a trainable logical network that accepts two logical inputs and can be trained to provide any logical function of the input variables. A block diagram of this device is shown in figure 1-1. This device is a direct implementation of the logical equation

$$C = \alpha_1 AB + \alpha_2 A\bar{B} + \alpha_3 \bar{A}B + \alpha_4 \overline{AB}$$

where A, B represent the logical inputs and C the logical output. A given logical function is established by selecting a suitable set of values ($j = 0, 1$). These values correspond to the positions of the statistical switches shown in the figure. For example, the function

$$C = A\bar{B} + \bar{A}B$$

can be obtained by letting

$$\alpha_2 = \alpha_3 = 1, \quad \alpha_1 = \alpha_4 = 0$$

This corresponds to closing switches S_2 and S_3 and opening switches S_1 and S_4.

Initial studies lead to a more general concept termed SOBLN (Self Organizing Binary Logical Network). This technique provides a general trainable logical network (TLN) with N-inputs and M-outputs. The N inputs are treated by forming the 2^N minterms and directing the signals to 2^N statistical switches. In a manner similar to that used for the ARTRON, the statistical switch outputs are directed to an OR gate to form the output. The M outputs are formed by providing a set of (2^N) switches and an OR gate for each output.

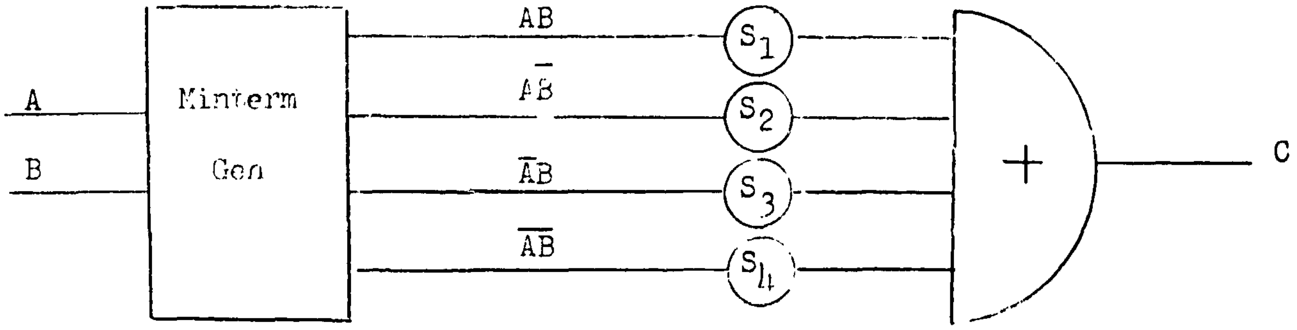


Figure 1-1 Two-Input, One Output TLN

The self organizing network is formed by the addition of a goal circuit. This circuit generates training signals termed reward/punish signals to the statistical switch in order to bias the switch probability of closure. Inputs to the goal circuit are signals related to system performance. The goal circuit then directs the TLN organization to satisfy the given objective.

Analysis of the organization process is based on a state representation of the TLN (a finite state machine) resulting from the method of construction and a transition matrix resulting from the goal circuit function. The total process can be shown to be a Markov process and is developed in detail in Appendix B.

The previous work was largely directed to the training of the TLN to a logical connective. This work included development of the general theory as well as specific applications. The work under this contract is concerned with the application of the TLN to solve decision problems. The results of this work is presented in the subsequent sections.

[illegible]

[illegible] Systems

[illegible]

[illegible] variable associated with a given system is taken to [illegible] performance of that system. It is distinguished from other attributes of the system in that there is always a known "optimum" range [illegible] for the variable. Most frequently, interest is centered upon the observation and control of this performance. A <u>performance vector</u> is an ordered set of such performance variables. In this light, performance of an aircraft system could correspond to its vector r.m.s. deviations from a given flight trajectory. Likewise, the performance of an environmental control system could correspond to the partial pressure deviations from a given temperature dependent norm.

In general, performance is a function of various random variables. It is itself, therefore, a random variable. Hence, statistical techniques can be applied to establish the properties of this performance. Frequently, control must be indirect, involving prediction or estimation of performance, experimentation, and control decisions.

A block diagram of one such a performance control system is provided for illustration in figure 2-1. The system considered is shown involving man and machine. The system behavior is subject to control inputs which determine the performance at time t_n. Other attributes of the system can be measured at time t_n, designated as the measurement vector

$$\vec{x}(t_n) = (x_1(t_n), x_2(t_n), \ldots, x_r(t_n))$$

The basic assumption is that future performance at time t_{n+1} is some

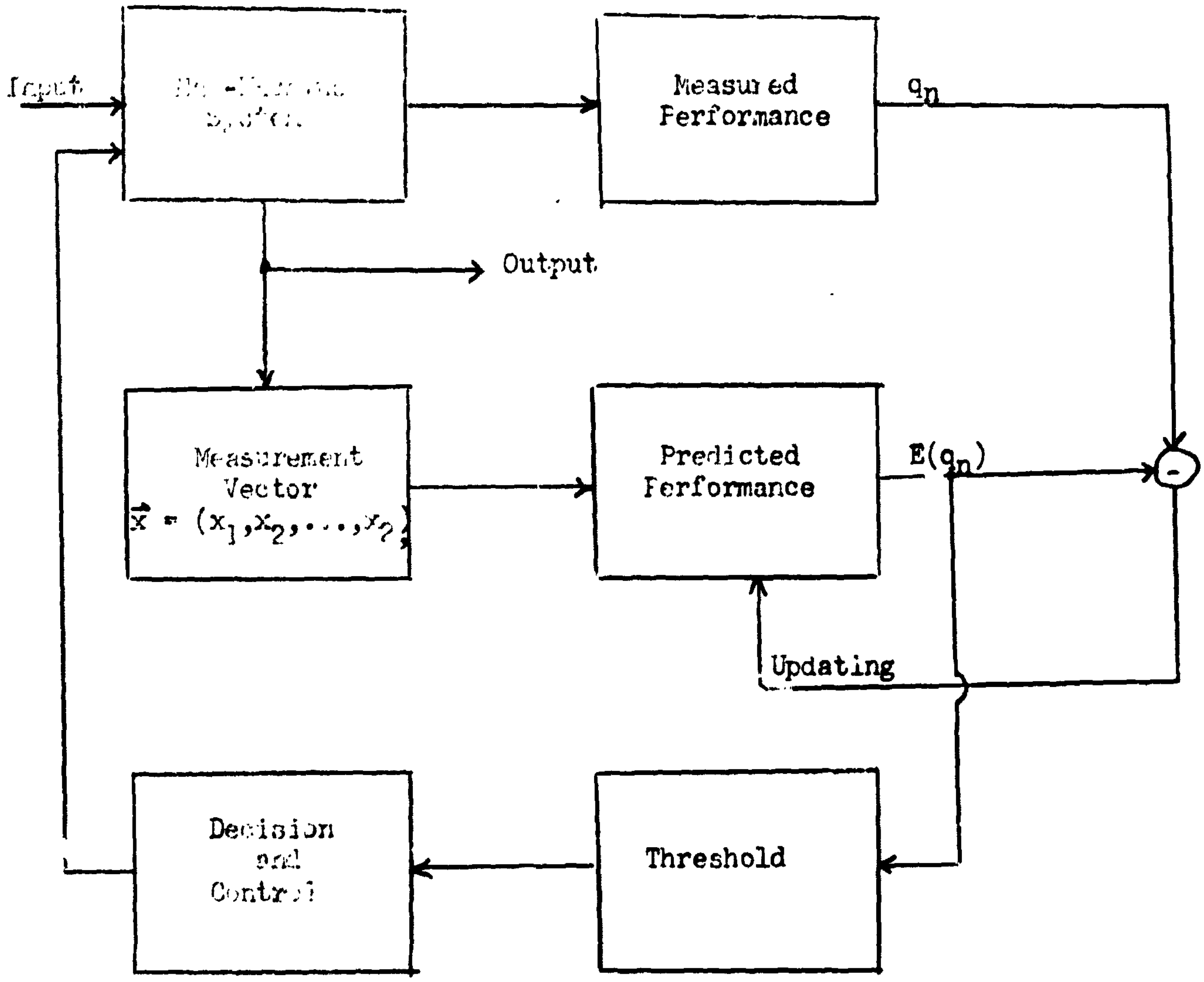


Figure 2-1 A Performance Control and Monitoring System (Illustrative)

function of these attributes at time t_n, $\vec{x}(t_n)$. That is, the assumption is that the performance variable can be predicted. The prediction technicue employed can be made adaptive, by an updating procedure shown as feedback to the performance prediction.

When the predicted performance falls out of tolerance, control determination experiments can be implemented. This would involve changing control parameters in such a way as to actually change the system parameters, $x(t_n)$, whereby the predicted performance would be changed. Modification would continue until predicted performance falls back within tolerance. A more probable approach would be to perform experiments which establish distributional properties of $x(t_n)$, restricting the class of modifications that are actually implemented. These can be selected on the basis of statistical theory.

The major elements of the above described performance control system involve adaptive prediction and decision theory. These will be considered in greater detail below.

2.1.2 Prediction and Estimation

Various techniques for establishing the performance prediction relationship

$$q_{n+1} = q\left[\vec{x}(t_n)\right]$$

have been developed and are described in the literature. One of these would consider performance, q, as being a discrete valued function; that is

$$q \in q^i \quad , \text{ for } i = 1,2, \ldots, n.$$

The r-components of the system measurement vector, $\vec{x}(t)$, would be viewed as the coordinates of a point in r-dimensional space. If one knows the distributions, $p(\vec{x}|q)$, the a priori probabilities, $p(q)$, and the loss matrix (corresponding to an estimate of the relative penalty associated with assigning performance q^i when q^j should be assigned) then the selection of q for a given measurement $\vec{x}$ can be made on the basis of minimum expected loss.(1)-(6) Other selection criteria can be used, however.

The technique is termed adaptive when either the required distributions $p(\vec{x}|q)$ or the loss matrix must be obtained from the incoming data. (The a priori probabilities, p(q), may or may not be known.) If measured-performance were noise free (i.e., the same as performance), then the problem becomes one of "supervised" learning. More generally, however, measured-performance is a random variable about whose distribution little may be known. This makes the problem here much more difficult.(7) Most of the work available, avoids some of this difficulty by assuming measured performance to be normally distributed. Furthermore, the form of the distributions $p(\vec{x}|q)$ is generally assumed as known. This last restriction could be removed in many cases, however.

Various alternative approaches can be applied to adaptive performance prediction. A familiar curve-fitting technique is described in Appendix A, as applied to prediction. Here, the performance, E(q), is represented by some given functional form

$$E\left[q(t_{n+1})\right] = f\left[\vec{x}(t_n), \vec{\Theta}(t_n)\right]$$

which is linear in $\vec{\theta}$, where $\vec{x}$ is the measurement-vector (an r-tuple) and Θ is a vector of unknown parameters (an S-tuple). These parameters are selected so as to minimize the sum of the squares of the deviations of measured from predicted performance over the discrete time-variable. If a fixed set of parameters $\vec{\Theta}$ were desired, the weights would be set to unity. Values less than unity permit time-variation in the parameters $\vec{\theta}$. The updating procedure is shown established by a very simple iterative process, and a well-known theorem applied to discussing the distributional properties of the parameter. This distribution over the predicted performance is important in forming control decisions, as will be seen in the next section.

2.2 Decisions and Decision Criteria

The simplest form of decision consists of selecting an action from a set of alternative actions with perfect information about the various consequences. Formally, one assumes a set of states-of-nature $\Omega = \{\omega_1, \omega_2, \ldots, \omega_r\}$ and a set of possible actions $A = \{a_1, a_2, \ldots, a_s\}$. A loss matrix is assumed, ((L(i,j))), whose elements L(i,j) are the loss associated with selecting action a_i while nature is in state ω_j. With perfect information, one knows both the state-of-nature and the loss matrix. A rational decision would be to select that action which yields the least loss.

With less perfect information, one might be restricted to knowing the cost matrix and only the a priori probabilities of nature being in state j, p (j), for j = 1, 2, ..., r (rather than the actual state of nature). In such a case one might make a selection on the basis of its yielding on the average, the minimum loss. That is, the expected loss associated with selecting action i is given by,

$$\mathcal{J}(i) = \sum_{j=1}^{r} L(i,j)\, p(j)$$

wherein one would select that action which minimizes $\int(i)$.

Consider now the case where one is given the loss matrix $((L(i,j)))$, but has no information on the probabilities over the states-of-nature. One method of establishing a decision is that employed in game theory. Here, one selects an action from a probability distribution over the available actions. This distribution over the available actions is established so that on the average, the maximum loss (sustained for any possible distribution over the states-of-nature) will be minimized.

A more useful class of decision problems extends the above considerations to include information obtained from experiments or observations. These are used to modify the established probability distribution over the states of nature. For example, let the observation be some parameter (or vector) $\vec{x}$. Let the conditional probabilities $p(\vec{x}|j)$ be known for each state-of-nature, j. Let the a priori probability that nature is indeed in state j, p(j), be also known. As before, let the loss matrix be given by $((L(i,j)))$, where the index i ranges over the set of possible actions and j ranges over the set of possible states of nature. For such problems, decisions are now based upon the observation $\vec{x}$. In fact, a decision rule is defined as any function mapping the observation $\vec{x}$ into an action i,

$$i = d(\vec{x})$$

The Bayes decision criteria (applied against the a priori distribution over the states of nature), is one that yields the minimum loss on the average. (That is, it is a criteria for selecting a decision rule which minimizes the average loss.) Its name derives from the use of Bayes theorem in probability which is used to derive these decisions. It can be readily shown[illegible] that this minimum average loss criterion implies that one should select action i when

$$\sum_{j=1}^{r} = L(i,j)\ p\ (\vec{x}|j)\ p\ (j) \leq \sum_{j=1}^{r} L(k,j)\ p\ (\vec{x}|j)\ p\ (j)$$

for all possible actions k. (This is essentially the criterion mentioned in discussing prediction and estimation in N-dimensional space.)

There are various other criteria that can be employed. One of these is the Neyman-Pearson criteria, as generalized to cover cases of more than two possible actions. With only two possible actions, say 1 and 2, the decision maker can hypothesize that action 1 is called for. He could then test this hypothesis and make two different kinds of errors. An error of the first kind would be made if his observation $\vec{x}$ led him to select action 2 when action 1 was called for (i.e., when the falsely rejected his hypothesis). An error of the second kind would be made if his observation $\vec{x}$ led him to select action 1 when action 2 was called for (i.e., when he falsely accepts his hypothesis). Whereas it is desired to minimize the probability of both of these errors, this is not in general possible. Normally, the decision rule which decreases the probability of one of these errors will increase the probability of the other type of errors. The Neyman-Pearson criteria calls for selecting that decision rule (function, d(x), which maps our observations into a selected action) which minimizes the probability of an error of the second kind, subject to the restriction that the probability of an error of the first kind remain below some pre-assigned value.[9] The generalization to the case of more than two actions can be accomplished in several ways.

In one of these (10,11), the probability of correct decisions is maximized while the probability of certain incorrect decisions are constrained to being less than or equal to some pre-assigned constants.

When more than one observation or experiment can be made, it becomes important to establish a criterion for stopping the process of experimentation (or observation) as well as the decision to be made once this has stopped. This is referred to as sequential decision theory and analysis.

Wald developed the sequential probability ratio test (SPRT)(12) for binary type decisions (accept or reject a hypothesis), terminating experimentation at some point beyond which a Neyman-Pearson type of criterion is satisfied. That is, numbers corresponding to acceptable maximum probability of errors of the first and second kinds are first selected. Experimentation ceases and a decision is made only when these conditions are satisfied by one of the two possible actions. His generalization of this test to multi-valued decision functions was made on the basis of minimizing the risk of making a wrong decision.

The Bayes sequential decision model postulates a given set of experiments (or observations), which can only be performed in the given order (i.e., experiment i must precede experiment i + 1). These experiments can be similiar to one another or completely different. If the set of experiments is finite, then it is called a truncated sequential theory. As with non-sequential Bayes decisioning, a loss matrix ((L(i,j))), a set of a priori probabilities, p(j), and a set of conditional probabilities, $p(\vec{x} | j)$ is presumed. The nature of the observation vector, $\vec{x}$, is that of including measurements made by all the experiments. Hence, decisions made on the basis of say n experiments can only use the conditional probability of observation-vectors whose first n-coordinates <u>only</u> are known. That is, one must average the expected loss over all coordinates corresponding to experiments which have not yet been performed. The last reouirement is to place a cost on each experiment which, in general, depends upon the outcome of the experiments.

The solution can be shown[8] to be obtainable by taking a dynamic programming type of approach and working backwards. Essentially, experimentation is to be continued only when the current Bayes risk (established as with non-sequential decisions to be average loss anticipated on the basis of current estimations of the state-of-nature) is greater than the expectation of loss if experimentation continues. The detailed solution is given in Appendix C.

For control or diagnostic purposes, this form of decision theory is not sufficiently general. Consider, for example, where a process is established as being in some one of three states (i.e., three states of nature) requiring immediate action. Three tests, labelled e_1, e_2, and e_3 are assumed available. Let these have costs C_1, C_2, and C_3, respectively. In addition, let results of experiments e_i provide coordinate x_i which is distributed as shown in figure 2-2. The expected loss and actual decision will depend upon the order in which these experiments are performed. For example, if experiment e_1 were performed first and provided the observation $x_1 = x_1^o$ (as shown) then one might cease experimentation and take that action corresponding to nature being in state $\omega = \omega_1$. If one obtained $x_1 = x_1^1$, then experiment e_2 would seem the

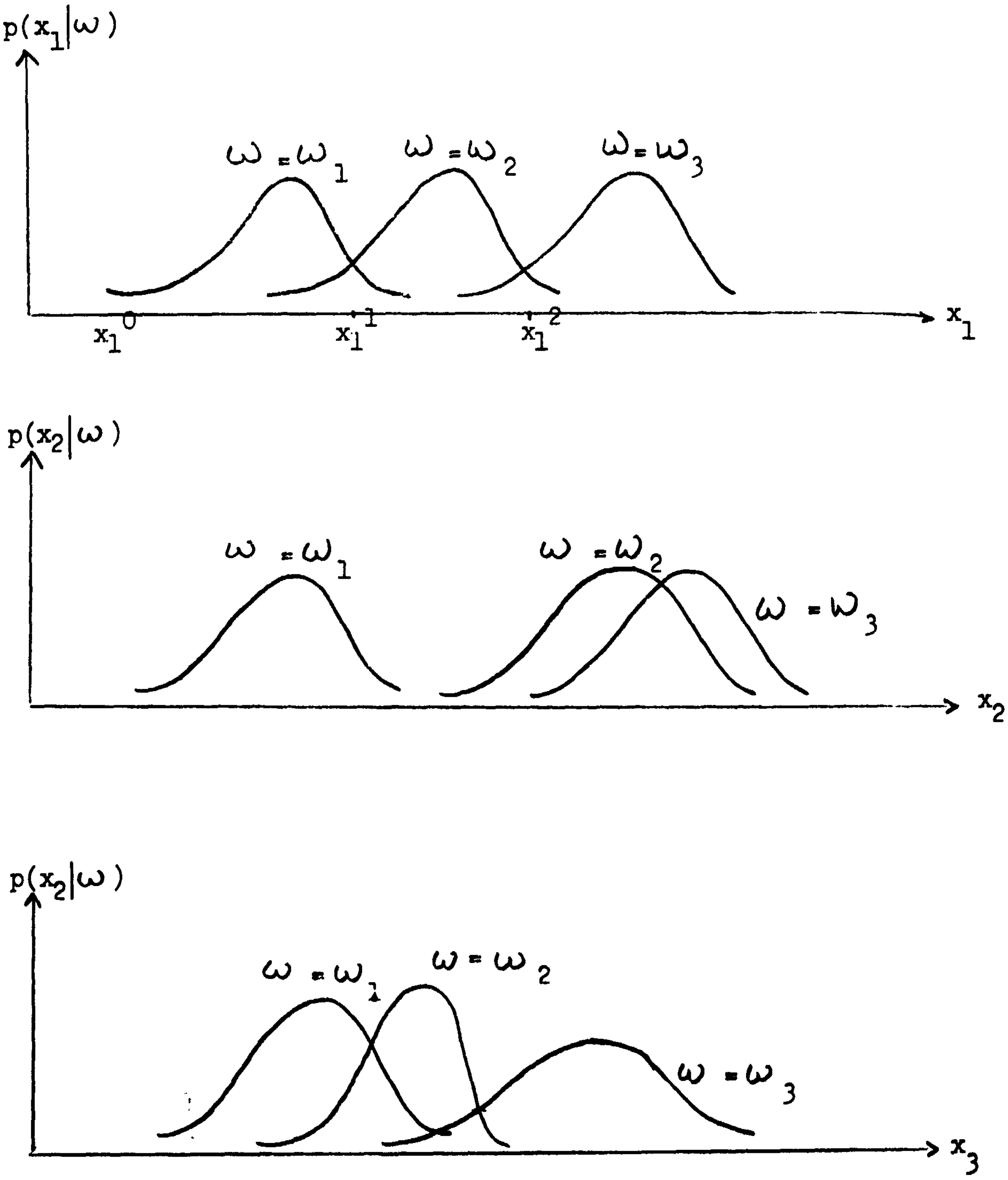


Figure 2-2 Hypothetical Conditional Distributions for States of Nature $\omega = \omega_1, \omega_2, \omega_3$

best experiment to perform next. If one obtained $x_1 = x_1^2$, then experiment e_3 might be best to perform next. Naturally, this would depend on the various loss elements assigned. (The example used here could correspond to problems of diagnosing the breakdown of a large system, where costs could be taken to represent time.)

This extension and generalization has been accomplished at Melpar, and is described briefly in Appendix C. Again, the dynamic programming type of approach is employed. At each stage of experimentation, the risk associated with making a decision is compared with the risk associated with making a decision is compared with the risk associated with each of the various possible continuations (experiments). If the risk associated with making an immediate decision is less than that associated with the possible continuations, then action is selected on the basis of the available data. Otherwise, continuation proceeds with that experiment which does not provide minimal risk.

This new procedure includes considerations where the performance of an experiment modifies the conditional distributions associated with other experiments. This would be expected to occur in many cases where the experiments actually perturb the process under consideration. However, one should employ this procedure anytime the distributions are not known to be independent.

To illustrate the difference between fixed ordered sequential theory and variable ordered sequential theory, consider the following simple search problem. Let there be an object which is randomly placed into one of eight boxes (so that the probability of being in any of the boxes is 1/8).

Let experiments e_j, $j = 1, 2, \ldots, 8$, gives rise to results

$$x_j = 1, \text{ if object is in box } j$$
$$= 0, \text{ if otherwise}$$

Let these experiments cost one unit. Let experiment e_9 give

$$x_9 = 1, \text{ if object is in boxes 1 or 2}$$
$$= 2, \text{ if object is in boxes 3 or 4}$$
$$= 3, \text{ if object is in boxes 5 or 6}$$
$$= 4, \text{ if object is in boxes 7 or 8}$$

and let it cost 2.5 units. If the loss matrix is such as to require a correct answer (with zero loss), then the minimum average loss for any fixed order of experiments would place experiment e_9 last (although all experiments would terminate prior to this point). With variable ordering of the experiments, the optimal procedure calls for performing e_9 first. It would produce an average loss of only 3.5 units as contrasted with the fixed order minimum loss of 4 units. The important point illustrated here, is that the initial experiment called for by the variable ordered sequential theory is not obtained trivially by considering the results of any fixed orderings of the experiments.

2.3 Computational Techniques

2.3.1 Introductory Discussion

There are several areas in which trainable logical networks (TLN) apply to performance control and monitoring systems similar to that described in the preceeding section. Its utilization rests upon certain key properties of such networks. One property is that they are finite state devices whose only memory consists of what state it is currently in. Another property is that they can be configured to behave as stochastic devices which can be analyzed as a Markov process. It appears naturel, then, to consider their application to such computational techniques as Monte Carlo and Simulation.

Other uses of TLN, however, have been studied previously. One of these is their use in obtaining high reliability in systems. Another, is in their use as control elements. This latter work, although not described here, will be considered relative to its application to the study problem selected. This section will concentrate on the use of TLN's for computation.

The basic element of the TLN, referenced as SOBLN, is a k-level statistical switch. This is simply a switch which can attain any of k states. Each of these states corresponds to a probability of the switch being closed. It is this element which is fundamental to the computation process described below. The fact that these devices are so flexible, so amenable to high-reliability considerations, and consist of this common element (wherein it is amenable to concepts of microminiaturization), provides reason for the investigation of their utility as basic computing elements as well as their use in problems amenable to solution by other than Monte Carlo techniques.

2.3.2 Basic Arithmetic Operations Using Statistical Switches

There are several methods for implementing the statistical switch to perform the multiplication and division of two numbers. The first method consists of converting both numbers into proper fractions by a scaling operation. Each number is then associated with a probability setting of a k-level statistical switch. The outputs of the switches are sent through an AND gate (alternatively, the switches may be merely placed in series). Since an n-bit counter is the basic element of a statistical switch, the k-level switch is one that is capable of taking on $k = 2^n$ different probability settings.

A Monte Carlo process is initiated with some number of samples, N, taken for conveneince to be a power of 2 (say 2^m). The 1 outputs from the above referenced AND gate will increment an m-bit counter. For a large enough m, one would expect that approximately

$$P_1 P_2 = \left(\frac{\text{number of 1's present in the counter}}{2^{-(m-2n)}}\right).$$

P_i = bias setting

To obtain the original, one merely shifts the counter m-2n bit positions to the right. This is a scaling operation which, in effect, corresponds to multiplying by the square of the scale factor in the denominator. Since the switches are independent, the AND function is represented by

$$P(AB) = P(A)\,P\,(B)$$

The accuracy in this Monte Carlo computation can be analyzed on the basis of the variance of a binomial distribution, given as

$$\sigma^2 = NPq,$$

where

$$q = 1 - P,$$

and where N is the number of independent samples.

To illustrate this process, consider the following example: Let N = 1024. The product of $3 \cdot 5 = 15$ could be performed using k-level switches, where

$$k = 2^n = 2^5 = 32.$$

Then,

$$P(A) = \frac{3}{32}. \quad P(B) = \frac{5}{32}$$

$$P(AB) = P(A)\ P(B) = \frac{15}{1024}$$

Since m = 2n there would be no shift of the counter after the end of the N samples. In general, however, one would anticipate much larger values for m, which would require the above described scaling shift.

The division of a scaler by another scaler can be readily performed by modifying the k-level switch to include a decoder which resets the n-bit counter at any integer r where $r < 2^n$. Thus, for the operation a/r where $r \geq a$ we merely use the foreshortened counter k-level switch with reset occurring on the r^{th} pulse and with the bias level set at a.

The second method considered for division involves the use of a closed loop feedback arrangement which adjusts the statistical switch until it reaches an equilibrium about a mean, where the mean is

$$p(y) = \frac{1}{2p(x)} \ .$$

This switch is then multiplied with the dividend p(z) which is set in a third switch. The rule for generating the transition matrix, corresponding to the generation of the state with probability p(y), is determined by a combinational network such as the one shown in figure 2-3.

The output of the k-level switch, p(y), is fed in series to the switch, p(z), whose output will yield the quotient (modified by $\frac{1}{2}$), of the form

$$e = p(y)p(z) = \frac{p(z)}{2p(x)}$$

The product operation can also be performed with the k-level switch, k = n, shown as p(y) and one m-level switch in a serial fashion as illustrated in the following example. Let the product of the two numbers, say 3 and 15, be computed. Set the numbers in the denominators of two fractions with a power of 2 in the numerator. Using the logic shown in figure 2-3 one needs $p(x) \geq .5$. However, a different implementation could be used in parallel to handle $p(x) < .5$. One can then sense the correct output. Using only one k-level switch, we interrogate the first twice with a counter reset at 3 and the second with the reset at 5. This yields a probability of 1 being $p(1) = \frac{2}{3} \cdot \frac{4}{5}$, as desired.

From the above discussion of the AND rule, one has

$$P(y) = \frac{1}{2P(x)} = \frac{1}{2 \cdot \frac{2}{3} \cdot \frac{4}{5}} = \frac{15}{16} \ .$$

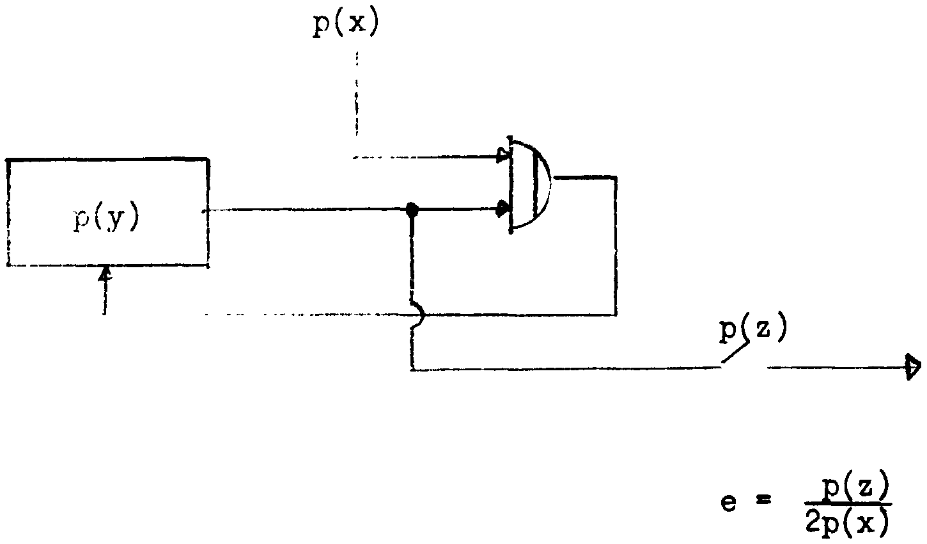


Figure 2-3 Stochastic Adjustment Rule

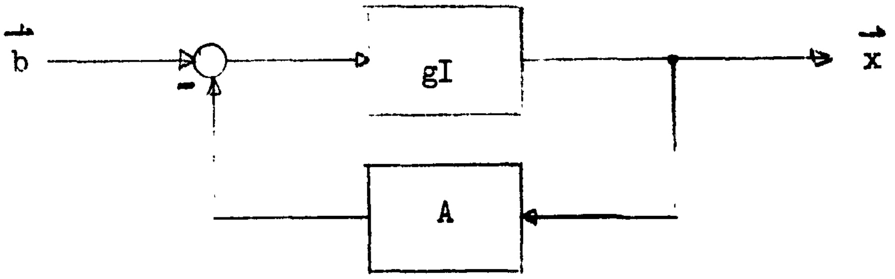


Figure 2-4 System for Solving Equations

Thus, when the equilibrium state is reached, the sampled value will be 15 with a variance which decreases with n.

2.3.3 Partial Differential Equations

One of the capabilities of a SOBLN as a Monte Carlo simulation device is in the solution of linear partial differential equations of the parabolic or elliptic type. We shall consider briefly only the parabolic type because of its relevence to random walk and diffusion processes. The Fokker-Planck equation, mentioned later, is a P.D.E. of the parabolic type.

Consider the general second order linear partial differential equation with two independent variables;

$$a(x,y)\frac{\partial^2\phi}{\partial x^2} + b(x,y)\frac{\partial^2\phi}{\partial x\partial y} + c(x,y)\frac{\partial^2\phi}{\partial y^2} + f(x,y,\phi_x,\phi_y) = 0$$

If (the discriminant) $b^2 - 4ac \leq 0$, the equation is of the elliptic type. Such equations commonly represent equilibrium situations; an example of which is the celebrated Laplace equation,

$$\frac{\partial^2\phi}{\partial x^2} + \frac{\partial^2\phi}{\partial y^2} = 0$$

Diffusion and heat flow equations are of the parabolic type with discriminant $b^2 - 4ac = 0$. They commonly represent situations with unbalanced equilibrium. Two examples of this are the one and two dimensional heat flow equations, described by

$$\frac{\partial u}{\partial t} = K\frac{\partial^2 u}{\partial x^2} ; \quad \frac{\partial u}{\partial t} = \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2},$$

respectively.

The standard approach to the one-dimensional expresion is separation of variables, yielding the solution

$$u = e^{c_1 t} \left[a \cosh t + b \sinh t \right]$$

A very useful means which exists for obtaining the solution of the two dimensional equations is the Monte Carlo process, where we consider a particle undergoing a series of random walks over a two dimensional lattice with a tallied score corresponding to the state after t transitions.(14) For an explanation of this process, consider the following situation. Let a particle undergo a series of random walks starting at (x,y) = (0,0) and continuing over the lattice for t steps with a transition probability at each point (x,y) associated with having the particle move to (x + 1,y), (x-1, y), (x,y+1), (x,y-1). Let each of these transition probabilities be equal. Assume, initially, that the transition probabilities are independent of x and y as well as the past history of the particle. This describes a Markov process with xy states and a symmetric transition matrix. The transition matrix has non zero terms along the two diagonals on either side of the main diagonal, where all non zero terms are $\frac{1}{4}$. The rest of the entries are 0.

At each point on the lattice there is a probability function P(x,y,t) of finding the particle at (x,y) after t transitions starting from (0,0). To show the relationship of this process to the heat equation, note that P(x,y,t) must satisfy the difference equation,

$$P(x,y,t+1) = \tfrac{1}{4} P(x+1,y,t) + \tfrac{1}{4} P(x-1,y,t) + \tfrac{1}{4} P(x,y+1,t) + \tfrac{1}{4}P(x,y-1,t)$$

This follows from the fact that the particle must have been at one of the above four positions at time t in order to arrive at (x,y) at time t+1. Subtracting P(x,y,t) from both sides and using the expression for second differences, we see that

$$\begin{aligned} \Delta f(x) &= f(x+1)-f(x) \\ \Delta^2 f(x) &= \Delta\left[f(x+1)-f(x)\right] \\ &= f(x+2) -2f(x+1) +f(x) \end{aligned}$$

Hence,

$$\begin{aligned} P(x,y,\ t+1) - P(x,y,t) &= P(x+1,y,t) - \tfrac{1}{2} P(x,y,t) + P(x-1,y,t) \\ &+P(x,y+1,t) - \tfrac{1}{2} P(x,y,t) + P(x,y\ -1,t) \\ &= \tfrac{1}{4}\Big\{ \left[P(x+1,y,t) -2\ P(x,y,t) + P(x-1,y,t)\right] \\ &\quad + \left[P(k,y+1,t) -2\ P(x,y,t) + P(x,y-1,t)\right] \Big\} \end{aligned}$$

This relates the first difference of P, with respect to t, to the second difference, with respect to x,y. For the limiting case of a finer lattice, the above difference equation is similar to:

$$\frac{\partial P}{\partial t} = K\left\{\frac{\partial^2 P}{\partial x^2} + \frac{\partial^2 P}{\partial y^2}\right\}, \quad K = \tfrac{1}{4}$$

This is the two dimensional heat flow equation.

If we keep a tabulation of the number of times the particle appears in each state (x,y) for a range of t and divide by the sample size, we have an estimate of P(x,y,t). If, instead of starting at (0,0), we start at a point on the lattice (x,y) where the starting point is determined by a distribution f(x,y), we have the initial function P(x,y,0) generating a particular solution of the difference equation.

By [illegible] the transition probabilities we can simulate other parabolic partial differential equations and, although the transition matrix becomes somewhat more complex, the analytical methods of section 3.2 and Appendix B can still be applied.

2.3.4 Matrix Inversion

A brief description is presented here of the inversion of a special type of matrix encountered when concerned with the control of Markovian processes. [illegible] procedure with a different implementation is possible, but less desirable, using the resolvent expansion of a matrix. In the section on application problems dealing with Markovian procedures the need for matrix inversion is avoided by means of using the iterative solution of a set of equations. Thus, only the storage of a vector is needed rather than a matrix.

We can invert a matrix of the form I-Q where Q is a stochastic matrix with all elements $q_{ij} \geq 0$ and where $\sum_j q_{ij} \leq 1$ for all i. To do this, we extend the dimension of Q by attaching a first row and a first column to it. These are selected so that the new matrix, A, will be stochastic. As an example, let

$$Q = \begin{bmatrix} \frac{3}{4} & 0 \\ 0 & \frac{3}{4} \end{bmatrix}$$

One then forms

$$A = \begin{bmatrix} 1 & 0 & 0 \\ \frac{1}{4} & \frac{3}{4} & 0 \\ \frac{1}{4} & 0 & \frac{3}{4} \end{bmatrix}$$

The matrix [illegible] will always be stochastic ($\sum_j a_{ij} = 1$ for all i and $a_{ij} \geq 0$) It describes a Markov process which is absorbing, owing to the selection of the elements of the first row. Because the eigen values of Q are less than unity, $(I-Q)^{-1}$ will exist and can be shown to equal $I + Q + Q^2 + \cdots$. Using analytic techniques such as the Z-transform, one can show (in closed form) the state probabilities for each transition, t. In reference (2) it is shown that these probabilities can be directly related to $(I-Q)^{-1}$. That is, using the above defined Q, one obtains

$$(I-Q)^{-1} = I+Q+Q^2+\cdots = \sum_{t=0}^{\infty}\left(\frac{3}{4}\right)^t \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} 4 & 0 \\ 0 & 4 \end{bmatrix}$$

This illustrates that we can allow a series of random walks to occur, where each time the absorbing state is reached the process is re-initialized. A TLN (similar to LANNET, for example) can utilize 2^N statistical switches as memory devices, and thus have a capability of handling a matrix of dimension N.

The random walk can progress in either of two ways. For the first method, let one K-level statistical switch represent the transition matrix with a single training rule generating the matrix. This configuration is feasible for lower order matrices possessing a large amount of symmetry and small number of non zero elements, such as the class discussed in section 2.3.3. An alternate method is to include N extra statistical switches j = 1,2,...N to this TLN. This arrangement allows each of these switches to represent a state of the Markov process considered and to have a separate training rule applied to each switch. These are called "state-switches". Such a training rule is merely a probability distribution representing the respective row of the transition matrix and is easier to synthesize.

The random walk can proceed over this set of N switches, (the remaining N^2 switches are used only as tallies). A decoder can route a tally bit to the proper memory switch each time the process is in switch j. The switches are associated with starting in state i and entering state k (switch (i,k)). These are activated when the random walk begins in state i and are incremented (tallied) when the corresponding "state-switch" enters state k.

2.3.5 One Method of Solving a Set of Simultaneous Equations

Consider the algebraic system of equations A $\vec{x} = \vec{b}$. Referring to the implementation of figure 2-4 one obtains the relation

$$\vec{x} = gIb' - AgIx$$

or

$$b = (A + \frac{1}{g})\, x$$

in the feedback loop.

Thus

$$\vec{x} = (A + \frac{I}{g})^{-1}\, \vec{b}$$

$$= \frac{\text{adj}\ (A + \frac{1}{g} I)\ \vec{b}}{\left| A + \frac{1}{g} I \right|}$$ See ref. (16)

for $\mu \to \infty$ (infinite gain in the amplifiers). Hence, the system in figure 2-4 solves the set of equation

$$A\vec{x} = b.$$

In this situation the stability of the system is guaranteed if matrix A is positive definite, since all characteristic roots would then have negative real parts. At the expense of some additional computation one can proceed in a similar manner when A is not positive definite. In such cases, we pre-multiply A by A^T, obtaining

$$A^T A \vec{x} = A^T \vec{b}$$

This has the same solution vector as the equation

$$A \vec{x} = \vec{b}.$$

However, the multiplication of two matrices is rather cumbersome for use by TLN.

2.3.6 Linear Difference and Differential Equations

We now consider the stability of an autonomous system of linear differential or difference equations. The system could be a vector-matrix state space representation of an $n^{\underline{th}}$ order linear difference equation. In the Markov Decision Process application, described in section 3.2, the Jacobi point iterative method of computing an optimal policy results in the following equation;

$$\vec{d}_k = F^k \vec{d}_o \quad \text{where } \vec{d}_o = \vec{x}_1 - \vec{x}_o, \; \vec{d}_k = \vec{x}_{k+1} - \vec{x}_k$$

and where $\vec{d}_k$ represents a vector difference which converges to zero as the iterative process converges to a solution. This convergence can only be guaranteed when the latent roots of the matrix F are ≤ 1. To show this, consider the difference equation related to the above expression

$$\vec{d}_{k+1} = F\vec{d}_k \quad \text{with given initial state } \vec{d}_o.$$

If the latent vectors are independent, we can apply a similarity transformation to diagonalize F, as

$$F = P\, D\, P^{-1},$$

where

$$D = P^{-1}\, F\, P$$

$$= \begin{bmatrix} \lambda_1 & & & & 0 \\ & \lambda_2 & & & \\ & & \cdot & & \\ & & & \cdot & \\ 0 & & & \cdot & \lambda_n \end{bmatrix} \qquad \lambda_i = i^{\underline{th}} \text{ latent root of } F$$

Since

$$F^2 = (P\, D\, P^{-1})\, (P\, D\, P^{-1}) = PD^2\, P^{-1}$$

one notes that

$$F^k = PD^k P^{-1}$$

$$= P \begin{bmatrix} \lambda_1^k & & & \\ & \lambda_2^k & & 0 \\ & & \ddots & \\ 0 & & & \lambda_n^k \end{bmatrix} P^{-1}$$

Thus, for large k, $\vec{d}_k = F^k \vec{d}_0 \rightarrow 0$ and the process converges to a solution. If the characteristic vectors are not independent then the matrix can always be transformed into a triangular matrix:

$$\vec{d}_k = R\,\vec{y}_k$$

or

$$\vec{y}_k = R^{-1}\,\vec{d}_k$$

Then

$$R\,\vec{y}_{k+1} = FR\,\vec{y}_k$$

or

$$\vec{y}_{k+1} = R^{-1} FR\,\vec{y}_k,$$

where

$$D = R^{-1} FR$$

is a triangular matrix.

Or, we note that we can always write D in the form where

$$D = D_1 + D_2$$

D_1 is a diagonal matrix and D_2 is a nil potent matrix (having non-zero elements only to the right of the main diagonal). Expanding D^p by the binomial theorem gives

$$D^p = D_1^p + PD_1^{P-1} D_2 + - - - + D_2^P$$

Since D_2 is nilpotent (all characteristic roots are 0), there exists a "p_2" such that $D_2^{p2} = 0$. It is also evident that for the diagonal terms in D_1, there exists some p_1 whereby D_1^{p1} will also be zero.

Other considerations apply to the continuous time case. Here, the set of differential equations

$$\dot{\vec{x}} = A \vec{x}$$

has solution

$$\overrightarrow{x(t)} = e^{At} \overrightarrow{x(o)}$$

$$= \left(I + A + \frac{A^2}{2!} + \frac{A^3}{3!} + - - -\right) \overrightarrow{x(o)} \qquad (17)$$

which can be evaluated in a manner described in the literature.

2.3.7 <u>Computation of the Inverse of the Least Square Recursion Formula</u>

In the recursion relation for updating a least squares estimate, described in the preceeding section, an expression of the form

$$P_{k+1}^{-1} = P_k^{-1} + \vec{\alpha}\,\vec{\alpha}^T \qquad\qquad P_k \text{ is known}$$

is encountered, where P_k^{-1} is an n by n symmetric matrix and $\vec{\alpha}$ is an n (13) by 1 column vector. Using a matrix inversion lemma, we can represent the above as

$$P_{k+1} = P_k - P_k\vec{\alpha}\,(\vec{\alpha}^T P_k\vec{\alpha} + 1)^{-1}\,\vec{\alpha}^T P_k$$

The above expression is possible to handle on a Monte Carlo basis since the underlined vector-matrix product $\vec{\alpha}^T P_k$ appears twice and since $P_k\vec{\alpha}$ is the vector transpose of $\vec{\alpha}^T P_k$. Thus, TLN's could perform a single vector-matrix product in the fashion described earlier in this report. Then, a dot product operation is performed to yield the expression $(\vec{\alpha}^T P_k)\vec{\alpha}$, using 2n statistical switches in the fashion identical to the first portion of a vector matrix computation.

After incrementing this result by 1, the resulting scalar $(\vec{\alpha}^T P_k\vec{\alpha} + 1)^{-1}$ is set into a single statistical switch consisting of a mod P counter (described earlier) with numerator 1. The bias probability is $\dfrac{1}{\vec{\alpha}^T P_k\vec{\alpha} + 1}$.

This switch is placed in series with all switches in the TLN to obtain the desired quantity

$$A = \frac{1}{\vec{\alpha}^T P_k\vec{\alpha} + 1}\; P_k\vec{\alpha}\,\vec{\alpha}^T P_k \;.$$

The quantity $(P_k\vec{\alpha})\,(\vec{\alpha}^T P_k)$ is obtained by a vector-vector product resulting in a matrix. The elements of this matrix

$$((a_{ij})) = (P_k \xi)\ (\xi^T P_k)$$

can be stored in the switches themselves to minimize the amount of the input-output logic. Since

$$P_{k+1} = P_k - A$$

we can subtract the elements P_{ij} from a_{ij} and change its sign, this operation being performed by a combinational network at each switch.

This technique, combined with the matrix inversion method described earlier, is sometimes a useful supplement to the standard schemes for solving a system of equations.

3.0 APPLICATION PROBLEMS

3.1 Trainable Controller, Problem 1

3.1.1 Problem Statement

Given that failures and/or changes in plant characteristics have occurred in an automatic control system, can trainable logic be designed to take over the control function by monitoring of human performance on manual control of the system?

3.1.2 Problem Motivation

Space flights to date indicate that man is one of the most reliable components in the complex man-machine system during space flights. In the future it is logical to expect that his capability to monitor space vehicle systems, perform control functions, and troubleshoot, make him a utility backup for many existing subsystems in the spacecraft. With this in mind, it seems reasonable to assume that his work load will vary greatly depending on how well things are going. Further, it seems reasonable to assume that the performance on tasks deteriorates if he becomes overloaded with work. It is possible for trainable logic to relieve some of the burden.

Let us suppose that one of the many automatic control or regulator systems fails and must be controlled manually. It is entirely possible that, while the failed automatic system is being controlled manually, trainable logic monitors both the astronaut's control and the data upon which the astronaut is basing his control decisions. After a while the trainable controller can signal that it is ready to make control policy. It is possible, and under some conditions probable, that the trained controller will perform better than the human which it has monitored.

3.1.3 Mathematical Formulation

We may set forth the following framework of the problem in a general state notation.

Let $\bar{x} = (x_1\ x_2, \ldots x_n)$ be metered system state variables,

$\bar{u} = u_1\ u_2, \ldots u_m$ be controller policy vector,

$\phi\ (\bar{u}) \leq 0$ a controller constraint,

$G(\bar{x},\bar{u},t) = 0$ a relation between control policy and system variables, and

$P(\bar{x},\bar{u})$ a performance index which is minimized by proper selection of $\bar{u}(\bar{x})$

Performance generally is marked:

$P(\bar{x},\bar{u}) < 0$ satisfactory

or

$P(\bar{x},\bar{u}) \geq 0$ unsatisfactory.

For the case we wish to study we may assume that a policy $\bar{u}(\bar{x})$ has been predetermined such that $P(\bar{x},\bar{u}) < 0$ until a controller failure or plant characteristic change occurs. In the latter case it is necessary to determine a new control policy $\bar{u}^*(\bar{x})$ such that $P(\bar{x},\bar{u}^*) < 0$. The new policy is simultaneously determined by the human and transferred to the trainable computer (controller). Additionally, it should be expected that the trainable logic gives some indication to the human when it is ready to take over control.

To give more meaning to the general framework let us specify parameters, constraints, plant equations, costs, etc. Let the plant be a servo motor which is adjusting to command inputs which are step functions. By letting the time intervals between step changes be much greater than the system time constant, the steps can be considered independent in time.

Nature selects any one from a number of plant equations by selecting i and j in the governing differential equation

$$\ddot{y} + a_i\dot{y} = k_j \, u_k \, (\dot{y}, y)$$

After a selection of (i,j) the control problem is to choose k such that a performance index P is minimized. As a performance index let us arbitrarily select an index which conserves both fuel and time.

$$P = \int_{t=t_o}^{t=t_f} C|u| + 1 \;\; dt$$

where $t_f - t_o$ is the time required to bring the system output to the input command.

As a constraint on the controlling policy, let us assume

$$\phi\,(u) = |u| - 1 \leqq 0$$

and actual permissible values

$$u = (1,\ 0,\ -1).$$

Let

starting time	$t = t_o$,
input	$z = z_o$,
output	$y(t)$,
error	$e(t) = z(t_o) - y(t)$,
error rate	$\dot{e}(t) = \dot{z}_o - \dot{y}(t) = -\dot{y}(t)$ for step input,
control policy	$u_1 = u(\dot{e}, e)$, and

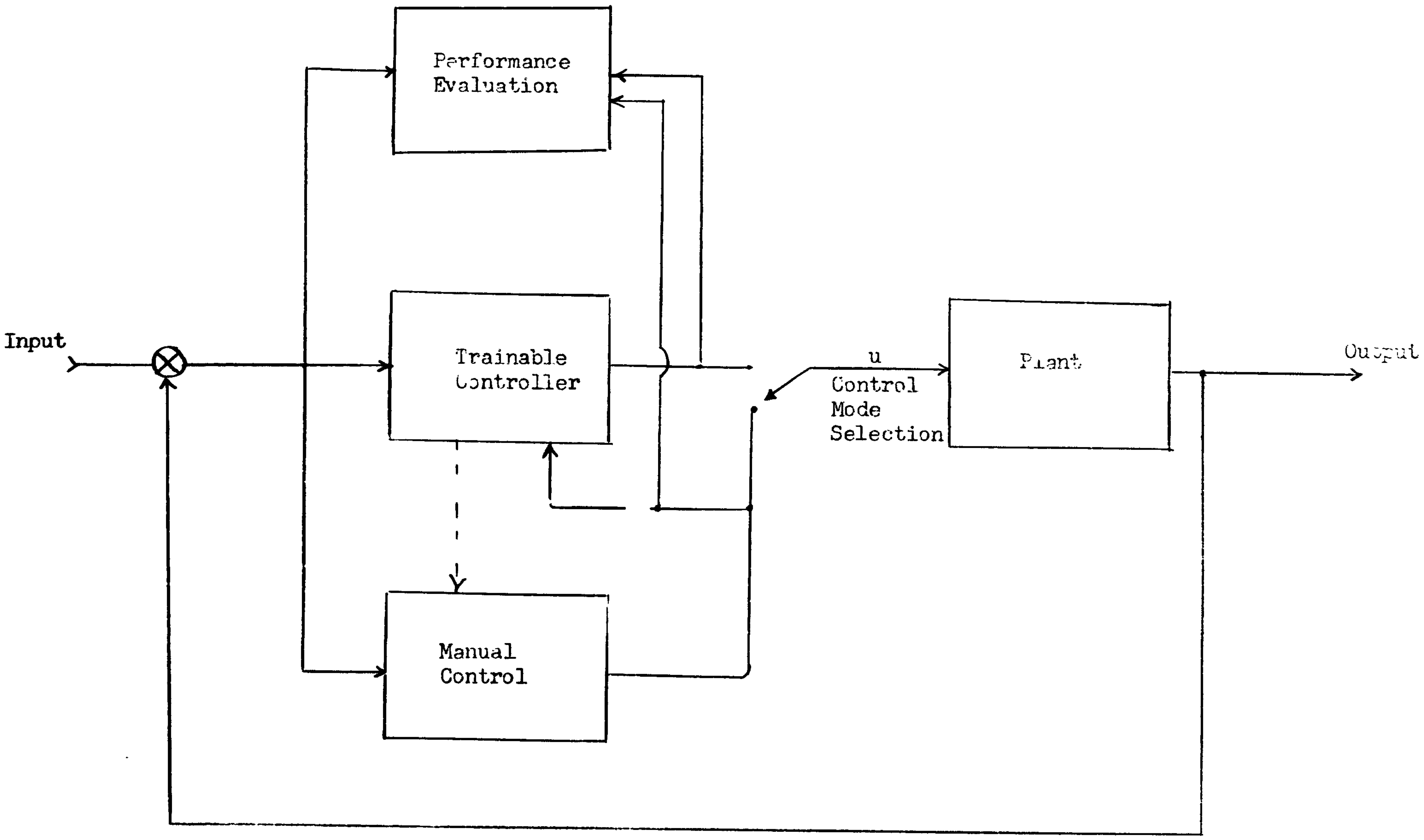


Figure 3-1 Control Policy Change with Adaptive Logic

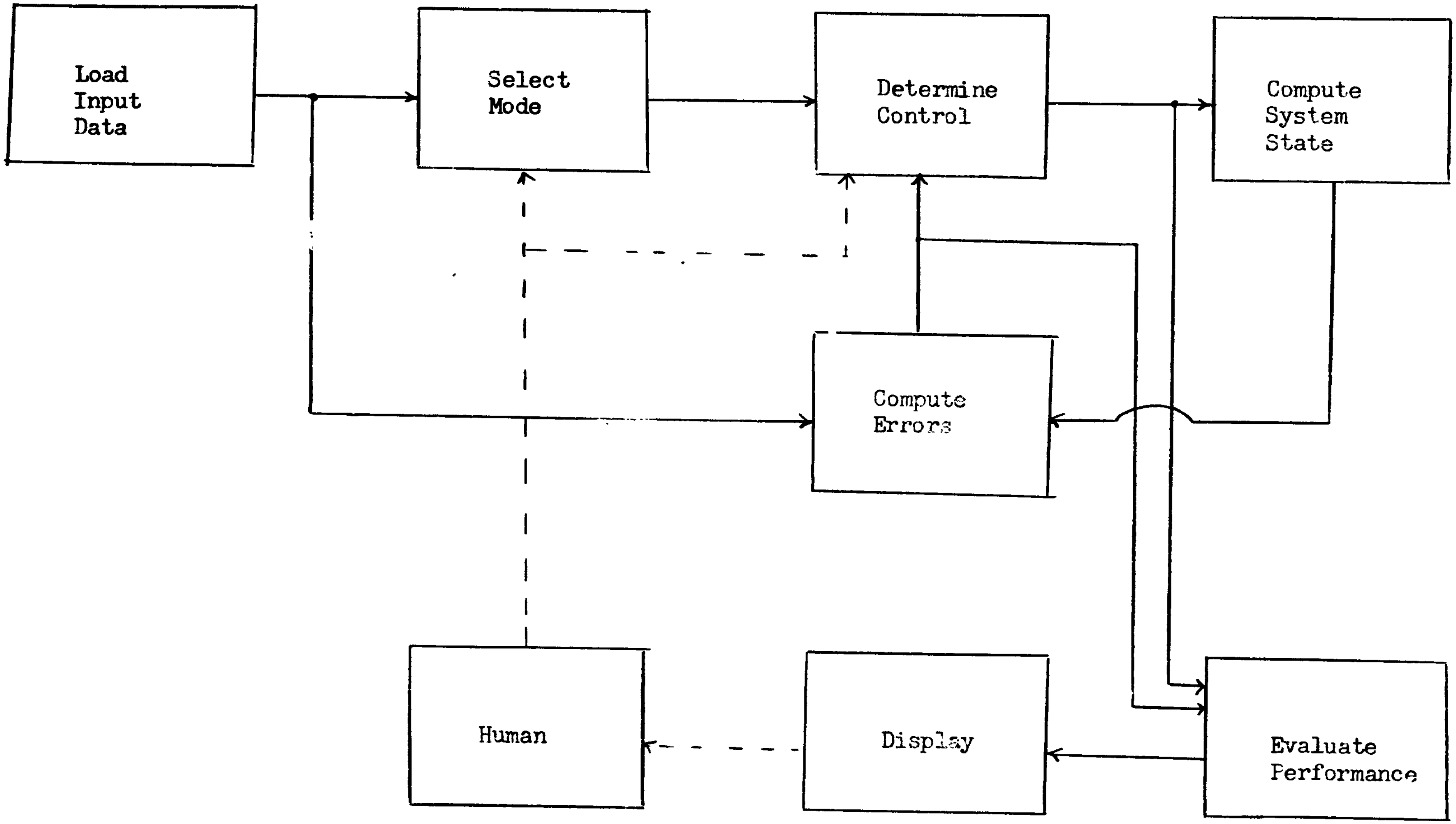


Figure 3-2 Flow Diagram - Simulation of Trainable Controller Problem

differential equation governing error be $\ddot{e} + a_1\dot{e} = k_1u_1.$

Starting at $t = t_o$ the above variables are:

$z = z_o,$

$y = y_o,$

$e = e_o,$ and

$\dot{e} = \dot{e}_o.$

At $t = t_f$, the error, the error rate and performance are:

$e = 0,$

$\dot{e} = 0,$ and

$$P_1 = \int_{t=t_o}^{t=t_f} c|u| + 1 \;\; dt$$

results from using $u = u_1$

A change in the desired policy control occurs when the coefficients (a,k) are not (a_1, k_1)

in the equation

$$\ddot{e} + a\,\dot{e} = ku.$$

3.1.4 Phase Plane Analysis

Let R_1, R_2, ... R_n be regions defined in the error-error rate plane. A control policy $u(\dot{e},e)$ is then determined by specifying a control value (1, 0, -1) for each region in the phase plane. Thus, for a given set of regions R_1, R_2 ... R_n all possible control functions could be enumerated.

Selection of a new control function by examination of all enumerated functions becomes impractical because of the large number of functions which are created by only a handful of regions. For example, with three control values and ten regions, there are 3^{10} control functions to choose from.

3.1.5 Control Policy Selection

The regions are defined by a set of binary variable. This, then, is the input space to the trainable logic. The output space is the set of control values (1, 0, -1). The mapping of regions to control values is accomplished by monitoring of the human operator who, though he may have no knowledge of phase space, is necessarily assigning values to regions as he controls the system.

As described, there appears to be no reason why the initial automatic control system should be distinct from the trainable controller. That is, the initial automatic control system will be the trainable controller trained to an initially specified control function.

It should be pointed out, there remains the problem of deciding when the trainable controller has received enough training to be trusted. Its reliability will depend on the variety of inputs it receives and the consistency with which the human operator behaves.

3.2 Markovian Process Control, Problem 2

3.2.1 Problem Statement

Given a man-machine system that is characterized by its being in a finite set of states, let the transition from one state to another state be representable as a stationary Markov process. Let the transition matrix, describing this operation, depend upon which of a finite set of policies (modes) the system is selected to operate under. We investigate the optimization of system performance through mode control.

3.2.2 Problem Motivation

There are any number of real and practical problems that can be represented in the above framework. Any system which is stationary Markovian (or can be approximated as such) and whose transition matrix is affected by some control parameter, gives rise to questions of the best parameter selection. To illustrate the applicability of such a development, various specific systems will be described in a later section.

3.2.3 Mathematical Formulation and Review

3.2.3.1 General Formulation

The theory required for solving the above problem has, for the most part, already been developed.(18) It is a logical extension of decision theory, and employs essentially the same theoretical foundations. The results, however, will be seen to be somewhat more sophisticated in certain respects.

A system is postulated, which exists in any of K-configurations. Let the system, in each of these configurations, be describable as a stationary Markov process. Control of the system is exerted through selecting any of the K-configurations. A control policy is a rule which determines the configuration to be selected. The problem posed is that of selecting an optimal control policy.

In order to solve this problem, the meaning of "optimal" in regard to such a system must be considered. This is accomplished by assigning costs and minimizing the expected loss.

Let each of the K-configurations involve M transition states. That is, let the transition matrix for any of the configurations be an M x M matrix. Assign a cost vector (for each of these matrices), giving the cost of being in any of these M states. Hence, for configuration i, we have the M x M transition matrix, A_i, and the N x 1 cost vector, c_i. Now, as the system proceeds through the various states (for any control policy) a cost is accumulated. The optimal control policy is that which minimizes the expected loss over (1) an infinite number of time increments, (2) a finite number of time increments, or (3) until the system attains a desired state.

3.2.3.2 Use of the Z Transform

The Z transform has been found to be useful in the analysis of Markov processes. Hence, before proceeding with the development of the optimum control of the above process, we shall briefly review the application of Z transforms to the analysis of Markov processes.

The Z-transform is defined as[19]

$$Z\left[f(k)\right] = \sum_{t=o}^{\infty} f(k)z^{-k} = f(o) + f(1)z^{-1} + f(2)z + \ldots \qquad (1)$$

which for f(k) = 1 gives

$$Z(1) = \frac{1}{1-z^{-1}} \quad \text{for} \quad |z| \geq 1$$

since

$$Z\left[f(k+1)\right] = \sum_{k=o}^{\infty} f(k+1)z^{-k}$$

$$= z\left\{f(1)z^{-1} + f(2)z^{-2} + \ldots.\right\}$$

$$= z\left\{\sum_{k=o}^{\infty} f(k)z^{-k} - f(o)\right\}$$

one obtains the difference equation

$$Z\left[f(k+1)\right] = z\left\{P(z) - f(o)\right\}, \tag{2}$$

for $\qquad P(z) = Z\ f(k)$

and $\qquad |z| \geq R$

with

R_c the radius of convergence of f(k). This can be applied to the analysis of Markov processes represented by equations of the form

$$\overrightarrow{p(k+1)} = A^T\ \overrightarrow{p(k)} \tag{3}$$

where A is a stochastic (transition) matrix with $\sum_{i=1}^{n} a_{ij} = 1$ for all columns j, and $\vec{p}$ (k) is the distribution over the possible states at time index k. Letting $\overrightarrow{p(o)}$ be the initial probability vector, then after k transitions one has

$$\vec{p}(k) = Q^k \, \vec{p}(o) \quad , \tag{4}$$

where

$$Q \equiv A^T.$$

Using Z transforms, one can solve for $\vec{p}(k)$ in a fairly easy manner. From the above derived difference equation, we see that

$$z \left\{ P(z) - p(o) \right\} = QP(z)$$

wherein we obtain

$$P(z) = (I - z^{-1}Q)^{-1} \; P(o) \tag{5}$$

which converges for all $|z| \geq 1$

The inverse transform yields the general form of $\vec{p}(k)$ in terms of k and $\vec{p}(o)$.

To illustrate this technique, consider the following example.

Let

$$A^T = \begin{bmatrix} \frac{1}{2} & \frac{1}{3} \\ \frac{1}{2} & \frac{2}{3} \end{bmatrix} \qquad \text{with } P(o) = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$$

Then,

$$\vec{P}(z) = \begin{bmatrix} 1-\frac{1}{2}z^{-1} & -\frac{1}{3}z^{-1} \\ -\frac{1}{2}z^{-1} & 1-\frac{2}{3}z^{-1} \end{bmatrix}^{-1} \begin{bmatrix} 1 \\ o \end{bmatrix} = (I - z^{-1}Q)^{-1}$$

$$= \frac{1}{(1-z^{-1})(1-\frac{1}{6}z^{-1})} \begin{bmatrix} 1-\frac{2}{3}z^{-1} & \frac{1}{3}z^{-1} \\ \frac{1}{2}z^{-1} & 1-\frac{1}{2}z^{-1} \end{bmatrix} \begin{bmatrix} 1 \\ 0 \end{bmatrix},$$

which by partial fraction expansion becomes:

$$p(t) = \begin{pmatrix} \dfrac{1-\frac{2}{3}z^{-1}}{(1-z^{-1})(1-\frac{1}{6}z^{-1})} \\ \\ \dfrac{\frac{1}{2}z^{-1}}{(1-z^{-1})(1-\frac{1}{6}z^{-1})} \end{pmatrix}$$

That is,

$$p(z) = \frac{1}{1-z^{-1}} \begin{pmatrix} \frac{2}{5} \\ \\ \frac{3}{5} \end{pmatrix} + \frac{1}{1-\frac{1}{6}z^{-1}} \begin{pmatrix} \frac{3}{5} \\ \\ -\frac{3}{5} \end{pmatrix}$$

Taking the inverse transform, one obtains

$$\vec{p}(k) = \begin{pmatrix} \frac{2}{5} \\ \\ \frac{3}{5} \end{pmatrix} + \left(\frac{1}{6}\right)^k \begin{pmatrix} \frac{3}{5} \\ \\ -\frac{3}{5} \end{pmatrix}$$

where the vector on the left represents the equilibrium state of the process.

3.2.3.3 Establishing Expected Loss for a One Configuration System

We can consider the analysis of a one-configuration ergodic Markov system having a cost which is associated with each state of the system. That is, we consider a system with one transition matrix and one cost vector. After k transitions, a mean cost increase can be determined associated with each initial state i. Let $f_i(k)$ be the expected (mean) cost over k transitions, where the system started in state i. It follows then, that

$$f_i(k + 1) = c_i + \sum_j q_{ji} f_j(k) \qquad (6)$$

where c_i is the cost of being in state i for one time index. Therefore, we see that

$$\vec{f}(k + 1) = Q^T \vec{f}(k) + \vec{c} \qquad (7)$$

where

$$\vec{f} = (f_1, f_2, \ldots, f_m)$$

and where

$$\vec{c} = (c_1, c_2, \ldots, c_m)$$

Since $\vec{c}$ is independent of k, its Z-transform is given by

$$Z (c) = \vec{c}Z(1) = \frac{1}{1-z^{-1}} \vec{c} = \frac{z}{z-1} \vec{c} \qquad (8)$$

Therefore, if we let

$$\vec{F}(z) = Z \left[\vec{f}(k) \right],$$

then we obtain

$$z \vec{F}(z) - \vec{f}(o) = Q^T \vec{F}(z) + \frac{z}{z-1} \vec{c}$$

or

$$\vec{F}(z) = (z J - Q^T)^{-1} \vec{f}(o) + \frac{z}{z-1} (z I - Q^T)^{-1} \vec{c} \qquad (9)$$

If, for convenience, we take

$$\vec{f}(o) = o$$

then the above equation reduces to

$$\vec{F}(z) = \frac{z}{z-1} (z I - Q^T)^{-1} \vec{c} \qquad (10)$$

3.2.3.4 Establishing Expected Loss for a M-Configuration System, With a Given Control Policy

As described earlier, a control policy is a rule for selecting a system configuration. Let this rule depend only upon the state the system is in. Then, each control policy describes what is equivalent to a one-configuration system whose transition matrix is a composite of those associated with the M-configurations. That is, row j of this matrix is selected as the $j^{\underline{th}}$ row of the transition matrix for some one of the configurations. The expected loss for this control policy is then established as described in section 3.2.3.3 for a one-configuration system. The optimal control policy is then the one that yields the minimum expected loss.

Consider, now, some selected policy u (which may or may not be optimal). After a large number of trials, one expects the system to settle down to an equilibrium distribution over the system states. This would imply that one could establish a mean cost per transition, L, for this control policy. Indeed, it can be shown[(18)] that for large k,

$$\vec{f}(k) = k\vec{L} + \vec{v} \qquad (11)$$

where f(k) is the expected loss vector, L the mean cost per transition vector, and v some constant vector. The components of these vectors represent the starting state, and the time-index, k, represents the number of transitions away from this start. If the system is ergodic, then the components of L can all be shown equal.

Applying the results shown in equation (7) to equation (11) yields

$$(k + 1) L + v = Q^T \left[k L + v \right] + c \qquad (12)$$

Since Q is stochastic, this reduces to

$$L + \vec{v} = c + Q^T v \qquad (13)$$

Subtracting

$$(v_n, v_n, v_n, \ldots, v_n)^T = Q^T (v_n, v_n, \ldots, v_n)^T$$

from both sides of this equation yields

$$\vec{L} + \vec{w} = \vec{c} + Q^T w \qquad (14)$$

where

$$\vec{w} = (v_1 - v_n, v_2 - v_n, \ldots., v_{n-1} - v_n, 0)$$

are the starting costs relative to state n. Since $\vec{w}$ involves (n-1) unknowns, and since L involves only one unknown (the components being equal for the ergodic case considered here), these n-equations are sufficient to solve for both $\vec{L}$ and $\vec{w}$.

3.2.3.5 Establishing an Optimal Control Policy for an m-Configuration System

Bellman's principle of optimality can now be used to establish an optimum control policy. Let the m-configuration transition matrices be $A_1, A_2, \ldots, A_m$. The procedure for establishing the optimal control policy is now as follows:

a. Select an arbitrary control policy, u_1.

b. Evaluate $\vec{L}(u_1)$ and $\vec{w}\,(u_1)$.

c. Form the vectors

$$\vec{Z}_j(u_1) = \vec{c}_j + A_j \vec{w}(u_1)$$

for each configuration, $j = 1,2,\ldots, m$.

(Note that $\vec{c}_j$ is the cost vector for configuration j and that it has components c_{ij} corresponding to the cost of leaving state i.)

d. Define "min $\vec{Z}(u_1)$" as a vector whose $i^{\underline{th}}$ component is the same as that of $\vec{Z}_j(u_1)$, for some j, and which is the minimum over all the other $i^{\underline{th}}$ components.

e. Form the new control policy, u_2, as the one that selects configuration j when the system is in state i if the $i^{\underline{th}}$ component of "min $\vec{Z}(u_1)$" was obtained from $\vec{Z}_j(u_1)$.

f. Repeat this process, solving for $\vec{L}(u_2)$ and $\vec{w}\,(u_2)$, forming "$\vec{Z}_j(u_2)$", determining "min $\vec{Z}(u_2)$", and then establishing u_3. This process will converge rapidly to an optimal policy.[18] This can be assessed by noting the convergence of $\vec{L}$.

3.2.3.6 Policy Determination by SOBLN as a Stochastic Computer

To implement the preceeding computation process, we shall obtain the solution of a set of simultaneous equations as an iteration process, where the format used is a set of linear difference equations. The criterion for convergence to a solution becomes identical to that described in showing stability of a linear autonomous, $n^{\underline{th}}$ order plant. Because of its appropriateness to the problem discussed here, this section was not included under the section on computational techniques.

Consider the system of equations

$$A\vec{x} = \vec{b} \tag{15}$$

Let

$$A = E - H \tag{16}$$

where E is a matrix which is conveniently inverted. Then the above becomes

$$E\vec{x} = H\vec{x} + \vec{b}.$$

(20)

The iteration process consists of inserting an initial estimate vector $\vec{x}_0$ on the right and solving for $\vec{x}$ designating it x_1. Thus,

$$E\vec{x}_1 = H\vec{x}_0 + \vec{b}$$

Repeating this with $\vec{x}_1$ gives

$$E\vec{x}_2 = H\vec{x}_1 + \vec{b},$$

etc, so that in general we obtain

$$E \vec{x}_{k+1} = H \vec{x}_k + \vec{b} \tag{17}$$

Therefore,

$$\vec{x}_{k+1} = E^{-1} H \vec{x}_k + E^{-1} \vec{b} \equiv F \vec{x}_k + \vec{s} \tag{18}$$

where

$$F = E^{-1} H \tag{19}$$

and

$$s = E^{-1} b \tag{20}$$

Considering the vector difference

$$\vec{x}_{k+1} - \vec{x}_k \equiv \vec{d}_k,$$

interpreted as an error-vector. Since

$$\begin{aligned} \vec{x}_{k+1} - \vec{x}_k &= F \vec{x}_k - \vec{x}_k + s \\ &= F \vec{x}_k - (H \vec{x}_{k-1} + s) + s \\ &= F (\vec{x}_k - \vec{x}_{k-1}), \end{aligned}$$

it follows that

$$\begin{aligned} \vec{d}_k &= F \vec{d}_{k-1} \\ &= F^2 \vec{d}_{k-2}, \text{ etc,} \end{aligned} \tag{21}$$

wherein

$$\vec{d}_k = F^k \vec{d}_o \tag{22}$$

$$d_o = \vec{x}_1 - \vec{x}_o.$$

For large k, F^k is approximately $\lambda_1 \vec{\alpha}$ where $\vec{\alpha}$ is the eigen vector corresponding to the largest eigen value λ_1. The accumulated differences will converge to a solution if $\lambda_1 < 1$ (lie in the unit circle).

From equation 18, we obtain the iteration process

$$\vec{x}_o = s = E^{-1} \vec{b} \tag{23}$$

$$\vec{x}_1 = (I + F)\, s = \vec{x}_o + F \vec{x}_o$$

$$\vec{x}_2 = (I + F + F^2)\, s = \vec{x}_o + F \vec{x}_1$$

$$\vdots$$

$$\underline{\underline{\vec{x}_{k+1} \qquad = \vec{x}_o + F \vec{x}_k}} \tag{24}$$

This recursion relation affords us with a simple means of solving a system of equations. An intelligent choice of E insures convergence to the solution, using the method described in this section, or possibly one of the inversion procedures described in the computation section.

Return now to the problem being considered. The set of equations in the policy iteration routine (section 3.2.3.4) the computations have the following form:

$$\vec{L} + \quad = \vec{c}_n + \; . \; \vec{w} \qquad (25)$$

That is, in expanded form,

$$L \cdot \begin{pmatrix} 1 \\ 1 \\ . \\ . \\ 1 \end{pmatrix} + \begin{pmatrix} w_1 \\ w_2 \\ . \\ w_{n-1} \\ 0 \end{pmatrix} = \begin{pmatrix} c_1 \\ c_2 \\ . \\ \\ c_n \end{pmatrix} + \begin{pmatrix} a_{11} & a_{12} & \dots & 0 \\ a_{21} & \cdot\cdot & \cdot\cdot & 0 \\ & & & 0 \\ & & & \\ a_{n1} & \cdot\cdot & \cdot\cdot\cdot\cdot & 0 \end{pmatrix} \begin{pmatrix} w_1 \\ w_2 \\ . \\ w_{n-1} \\ 0 \end{pmatrix}$$

where the matrix used is derived from A by replacing its last column with zeros. This does not change the equation. Obviously, the eigen-values of this new matrix are less than 1. This equation can now be rewritten as

$$\begin{pmatrix} 1-a_{11} & -a_{12} & \dots & 1 \\ -a_{21} & 1-a_{22} & \cdot\cdot & 1 \\ & & & \\ & & & 1 \\ -a_{n1} & \dots\dots & & 1 \end{pmatrix} \begin{pmatrix} w_1 \\ w_2 \\ \\ w_{n-1} \\ L \end{pmatrix} = \begin{pmatrix} c_1 \\ c_2 \\ \\ \\ c_n \end{pmatrix}$$

or

$$A' \vec{w}' = \vec{c} \qquad (26)$$

Let E be the matrix:

$$E = \begin{pmatrix} 1 & 0 & 0 & . & . & . & 1 \\ 0 & 1 & 0 & . & . & . & 1 \\ . & . & . & . & . & . & . \\ 0 & 0 & . & . & . & 1 & 1 \\ 0 & . & . & . & . & . & 1 \end{pmatrix}$$

By inspection, we obtain its inverse as

$$E^{-1} = \begin{bmatrix} 1 & 0 & 0 & . & . & . & .-1 \\ 0 & 1 & 0 & . & . & . & .-1 \\ . & . & . & . & . & . & . \\ 0 & 0 & . & . & . & 1 & .-1 \\ 0 & . & . & . & . & . & .1 \end{bmatrix}$$

Letting

$$A' = E + H$$

we see that

$$F = - E^{-1} H = \begin{bmatrix} 1 & 0 & . & . & .-1 \\ 0 & 1 & 0 & . & .-1 \\ . & . & . & . & . \\ 0 & 0 & . & 1 & .-1 \\ 0 & . & . & 0 & 1 \end{bmatrix} \begin{bmatrix} a_{11}, a_{12} & . & . & a_{1,\ n-1,} & 0 \\ a_{21} & . & . & . & 0 \\ . & . & . & . & . \\ . & . & . & . & . \\ a_{r1} & . & . & a_{n,n-1} & 0 \end{bmatrix} \qquad (27)$$

That is,

$$F = \begin{bmatrix} a_{11}-a_{n1} & a_{12}-a_{n2} & . & . & . & 0 \\ a_{21}-a_{n1} & a_{22}-a_{n2} & . & . & . & 0 \\ . & . & . & . & . & . \\ & & & & & \\ a_{n1} & a_{n2} & . & . & . & 0 \end{bmatrix} \qquad (28)$$

where

$$a'_{ij} \geq 0 \ , \quad \sum_{j=1}^{n-1} a'_{ij} < 1 \qquad \text{for all } i$$

We can now substitute the above relation into equation 24 to implement the iterative procedure for policy determination.

3.2.3.7 Specific Systems for Application

As mentioned earlier, many systems in nature can be analyzed in the Markovian framework. It should be noted also that often, by a modest extension of the state space, natural processes which are otherwise not Markovian can be put into the Markov framework.

The techniques presented here could utilize TLN as either (a) a controlled process, or (b) as the computational element which determines optimum policy for control of a naturally occurring Markov process.

a. Using TLN as the controlled process, we have an analytical method for determining the effectiveness of various training rules in directing it to an absorbing state, with costs associated with temporary presence in each state during training, and costs associated with the use of each rule.

b. For the optimum control of Markov processes we use TLN as a stochastic computer to compute, on or off line, the control policy. For a regular, non-absorbing, process the theory allows us to compute a state dependent control policy which minimizes mean cost/unit time. (Fig. 3-3).

Listed below are some examples of possible Markov systems. It is likely that numerous examples (not included in the list) exist, for which investigation would be profitable.

1. Regular processes

a. Regulation of closed loop (linear & non-linear) control systems with random inputs. (Fig. 3-4)

b. Inventory problems with stochastic demand.

c. Regulation & control of ecological processes.

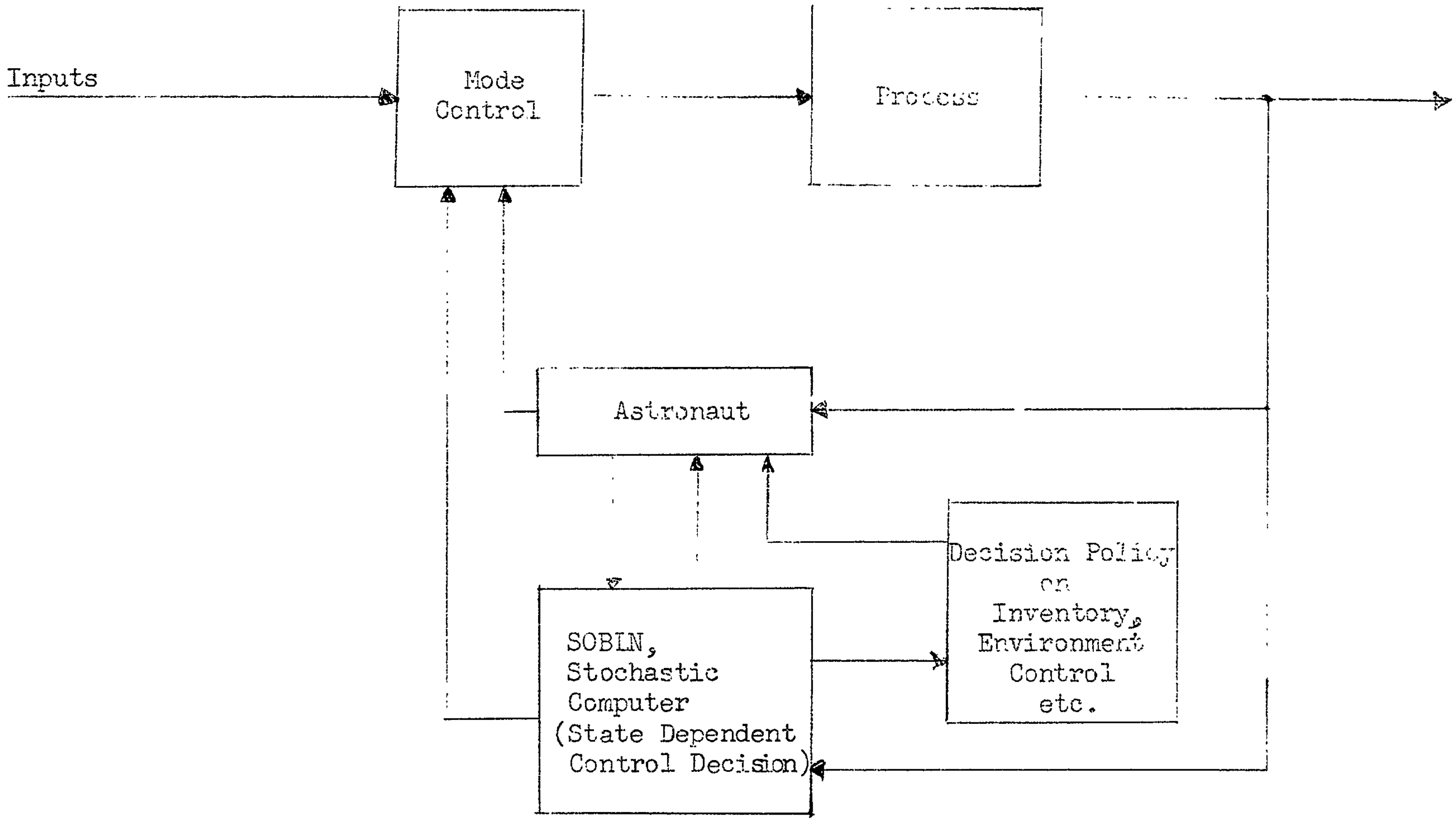

Inputs
Mode Control
Process
Astronaut
SOBIN, Stochastic Computer (State Dependent Control Decision)
Decision Policy on Inventory, Environment Control etc.

Figure 3-3 Block Diagram - Markov Decision Process

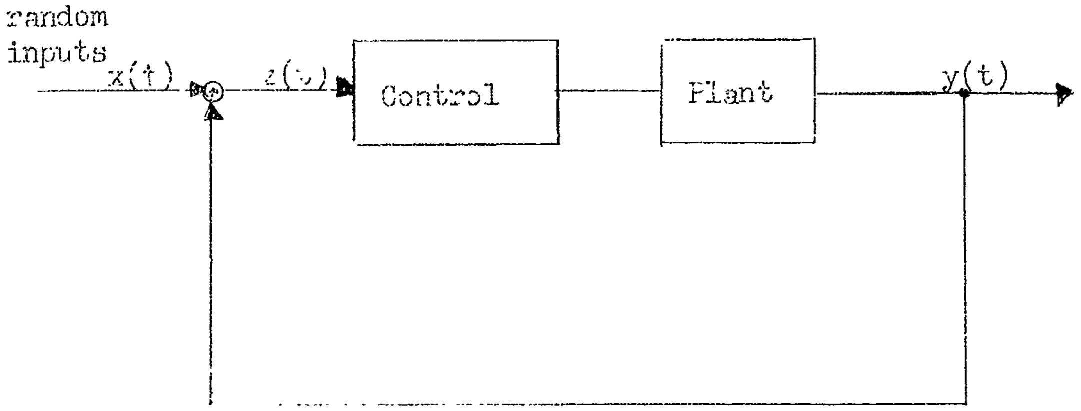


Figure 3-4 Closed Loop System with Random Inputs

2. Absorbing Processes

a. Control systems with random disturbances which lock on to the reference signal when within a prescribed error range (absorbing state).

b. If the reference signal is noisy, the above problem becomes one of computing an optimal pursuit trajectory.

c. Reliability analysis of systems with machine breakdown and replacement or repair.

3.2.3.8 Simulation Problem

For a simulation study, a regular Markov process will be simulated on the SDS-910 Computer in Fortran using random number generation. A set of control actions will be available each one of which governs the behavior of the process when its respective control action is being implemented.

The problem will be the determination, by TLN, of a control policy which will minimize the mean costs/time where the cost structure is represented by a cost vector associated with each transition matrix.

The ensemble of transition matrices could represent the response of the human to configurations and displays in the control system interface.

The TLN will compute an optimal policy (by Monte Carlo) based on the mathematical input data. (See Fig. 3-5.)

1. Objective

Measure speed and efficiency of TLN as a computational element for computing optimum policy.

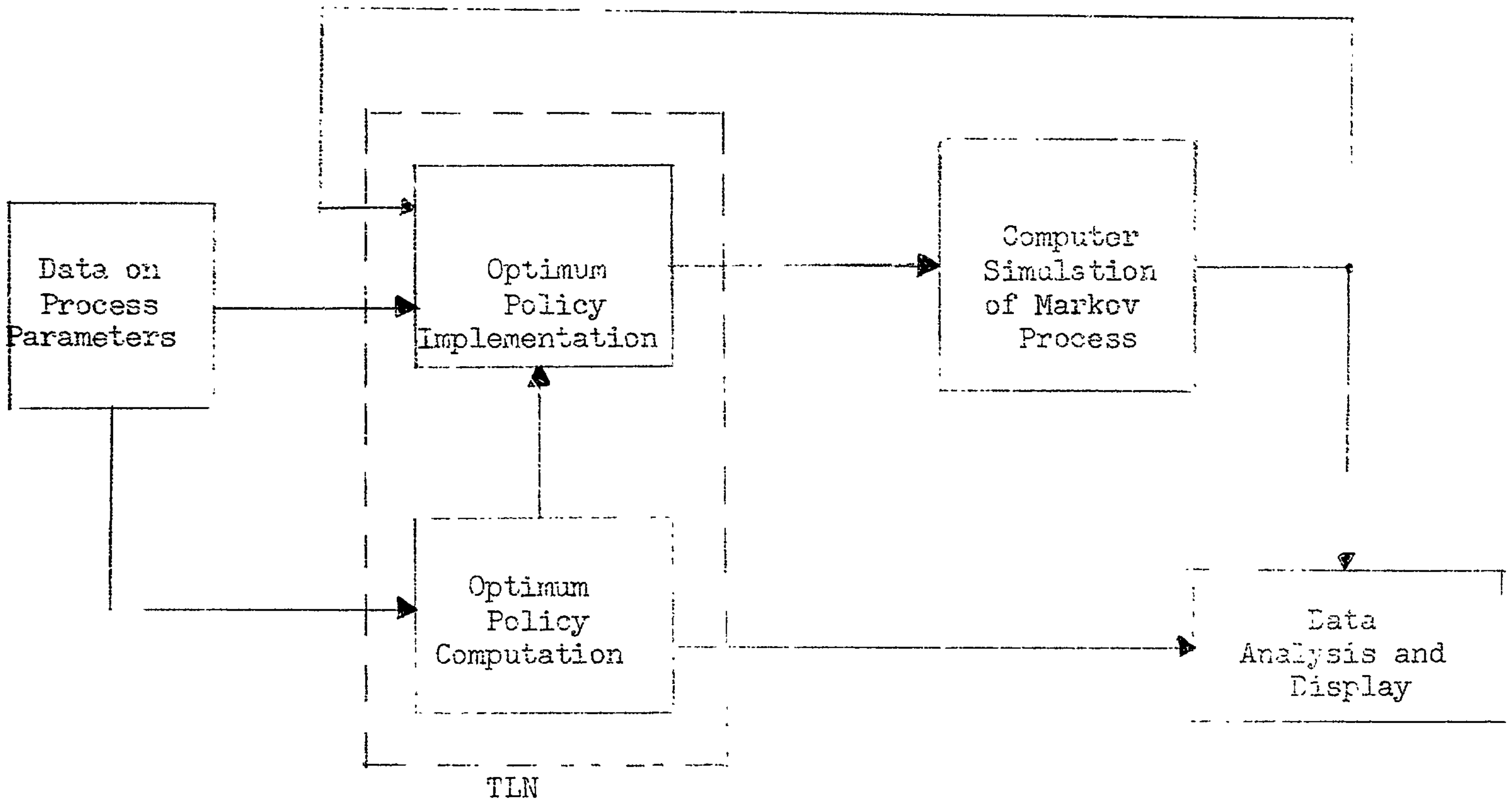

Data on Process Parameters
Optimum Policy Implementation
Optimum Policy Computation
TLN
Computer Simulation of Markov Process
Data Analysis and Display

Figure 3-5 Simulation Problem

3.3 Bayes Decision Making (Problem #3)

3.3.1 Problem Statement

This problem is the application of trainable logical networks (TLN) to the on-line solution of Bayes decisions. The specific decision problem is the routing of signals along one or more paths. As shown in figure 3-6 the signals are processed to form inputs acceptable to the decision computer.

The decision computer determines what output channels are to be activated according to the lowest cost Bayes criterion. One output path includes an external evaluation device that can modify the cost matrix contained in the decision computer.

3.3.2 Problem Motivation

As an example of the type of system treated consider the problem of processing medical and/or environmental data in a space vehicle mission. The various possible actions may be to telemeter to earth, record data, direct data to displays, and combinations of these and other actions.

For Bayes decisioning it is necessary to establish beforehand the probabilities that the process is in a certain state given the input signal. For instance, one method of classification of EKG waveforms involves the identification of four pathological states: inferior myocardial infarction, right bundle branch block, and left and right ventricular hypertrophy as well as the normal state.

The use of TLN as a decision performing device reruires two conditions.

(1) An estimate of costs of taking each action is known and is reasonably accurate. This cost matrix can be updated periodically by

the astronaut or ground command in the light of developments during a mission.

(2) A reasonably accurate estimate of the probability distribution over the states of nature (process). This data results from EKG or other data taken on previous missions, in simulation runs, on other personnel etc. It is reasonable to assume that sufficient data will be available to base a probabilistic estimate of the state of nature.

Two general methods of pattern recognition are possible, parametric and non-parametric techniques. The considerations above apply in general to parametric pattern classification techniques where the states of nature (pathological and normal conditions of the heart etc.) can be adequately described as population classes, each class represented by a mean vector and covariance matrix. An input can be classified on the basis of a set of probabilities as to which class it belongs.

Other data (such as life support parameters) representing scalar variables would be described by the mean and variance or other parametric information.

In the system block diagram figure 3-6 , the control functions may consist of other actions besides those listed above such as: obtain more information on the process (man or machine) by further measurement. Such an action may be the performance of a blood pressure test.

Feedback from ground control taking the form of the physicians conclusions drawn from telemetered data could form an adaptive loop for updating the distribution utilized as inputs to the Bayes decision device.

In the life support system the decision space can be thought of in several ways, one being different degrees of control of the environment

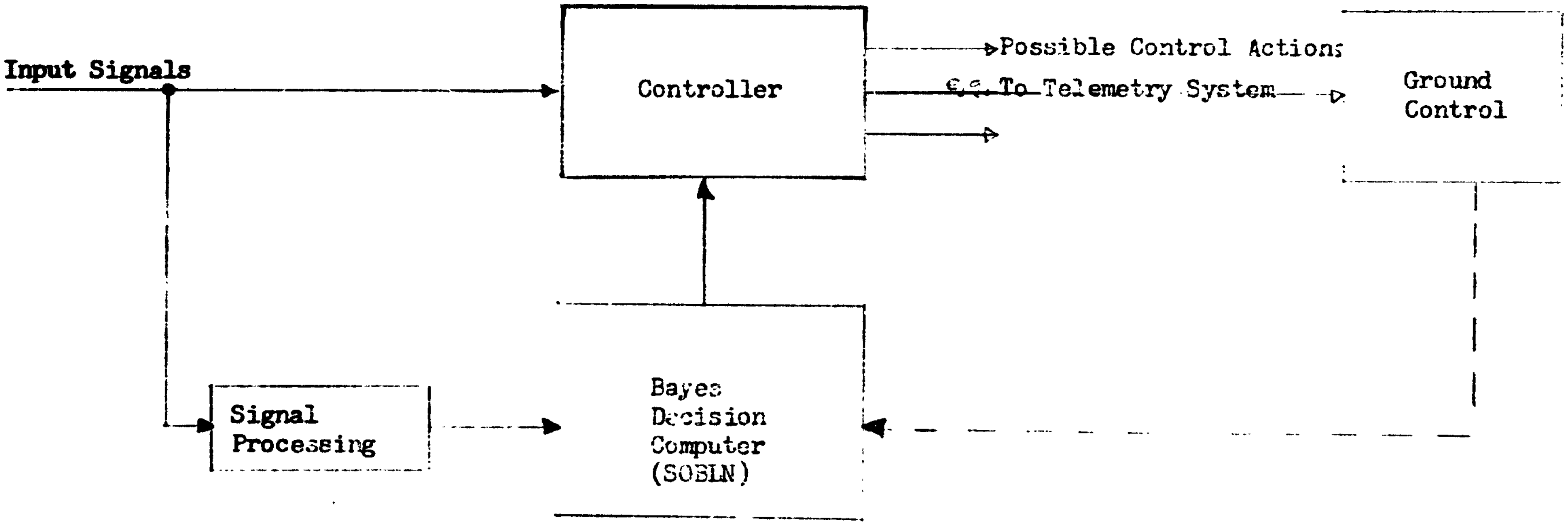


Figure 3-6 Block Diagram - Bayes Decision Problem

(such as temperature or oxygen flow rate control).

3.3.3 Theory of a Bayes Decision Against a priori data

The Bayes decision (strategy) consists of the application of a strategy vector $\vec{a}$ which produces:

$$\min_{\vec{a}} \quad \vec{e} \cdot M \vec{a}^{T} \, , \quad \sum_{i=1}^{m} e_i = 1 \, , \sum_{i=1}^{n} a_i = 1, \; a_i \geq 0$$

Where M is an n by m cost matrix and $\vec{e}$ is an assumed (a priori) distribution vector signifying the state of nature: the value of the Bilinear form shown above for a particular $\vec{e}$, $\vec{a}$ and M is the expected payoff of the decision.

$$\begin{bmatrix} e_1, & e_2, & \cdots & e_m \end{bmatrix} \begin{bmatrix} m_{11}, & m_{12} & \cdots & m_{1n} \\ m_{21}, & \cdots\cdots\cdots & & \\ . & . & . & . \\ m_{m1} & \cdots\cdots\cdots & & m_{mn} \end{bmatrix} \cdot \begin{bmatrix} a_1 \\ a_2 \\ . \\ . \\ a_n \end{bmatrix} = P$$

Thus we want to select $\vec{a}$ pure strategy $\vec{a}$ (a vector with all elements zero except for one element equal to unity) which will minimize the above expression given "a priori" knowledge of $\vec{e}$.

We can illustrate the concept best by considering the following example:

since $\displaystyle\sum_{i=1}^{2} e_i = 1$

let $e_1 = e$

$e_2 = 1-e$

$$\begin{bmatrix} e, & 1-e \end{bmatrix} \begin{bmatrix} 2 & 1 & 3 \\ 1 & 3 & 1 \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \\ a_3 \end{bmatrix} = \vec{e}\, M\, \vec{a}^T = R$$

where R is the rick and where $\vec{a}^T$ is the vector transpose of $\vec{a}$ from matrix algebra $A\,B^T = (B\,A^T)^T$.

Thus

$$\begin{bmatrix} e, & 1-e \end{bmatrix} \left(\begin{bmatrix} a_1 & a_2 & a_3 \end{bmatrix} \begin{bmatrix} 2 & 1 \\ 1 & 3 \\ 3 & 1 \end{bmatrix} \right)^T = \vec{e}\; M\; \vec{a}^T = R$$

$$\begin{bmatrix} e, & 1-e \end{bmatrix} \left(a_1 \begin{bmatrix} 2,1 \end{bmatrix} + a_2 \begin{bmatrix} 1,3 \end{bmatrix} + a_3 \begin{bmatrix} 3,1 \end{bmatrix} \right)^T = R$$

The linear combination of vectors (columns of M) shown above in parenthesis is commonly referred to in decision theory as a convex hull, we designate this vector $\vec{S}$.

Thus we wish to minimize the dot product $\vec{e}\, .\, \vec{S}$ through the selection of the best $\vec{a}$.

Thus for an assumed $e = \left[\frac{1}{2}, \frac{1}{2}\right]$ we have

$$\vec{S} = (1\begin{bmatrix} 2, & 1 \end{bmatrix} + 0 \begin{bmatrix} 1, & 3 \end{bmatrix} + 0 \begin{bmatrix} 3, & 1 \end{bmatrix}) \text{ or}$$

$\vec{a} = \begin{bmatrix} 1, & 0, & 0 \end{bmatrix}$ for the Bayes decision which yields min $\vec{e}\, .\, \vec{S}$.

The optimum $\vec{a}$ was determined by constructing the perpendicular to $\vec{e}$ and sliding it until it intersected the first vertex of the area $\vec{S}$,(Fig 3-7). The equation of the perpendicular is $\vec{e}\, .\, \vec{s} = C$. C is constant, thus the

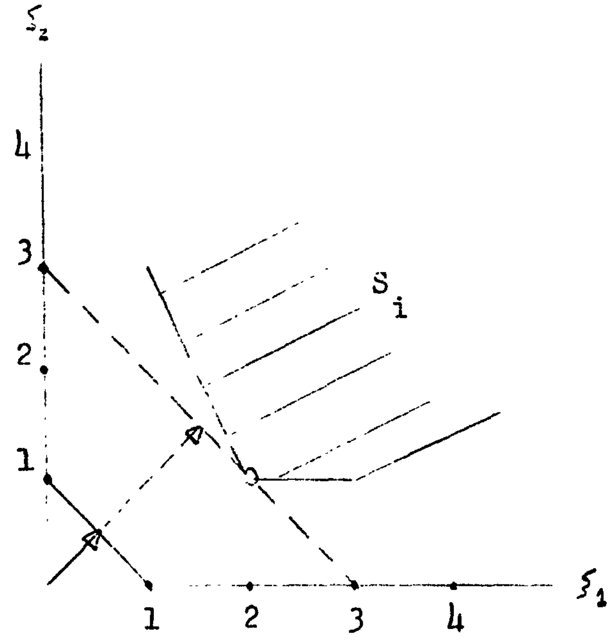


Figure 3-7 Construction of Bayes Decision

where C_{ij} are costs which range over $C \leq C_{ij} \leq 1$ (normalized to 1). We have d_i as the probability of a system being in the i^{th} state of nature thus $d_1 + d_2 + \cdots + d_i + \cdots + d_m = 1$.

The V_i then represent the Bayes risk function where V_i is the dot product of $\vec{d}$ and $\vec{C}_i = \left[C_{1i}, C_{2i}, \cdots C_{mi}\right]$

$$V_i = \vec{C}_i \cdot \vec{d}$$

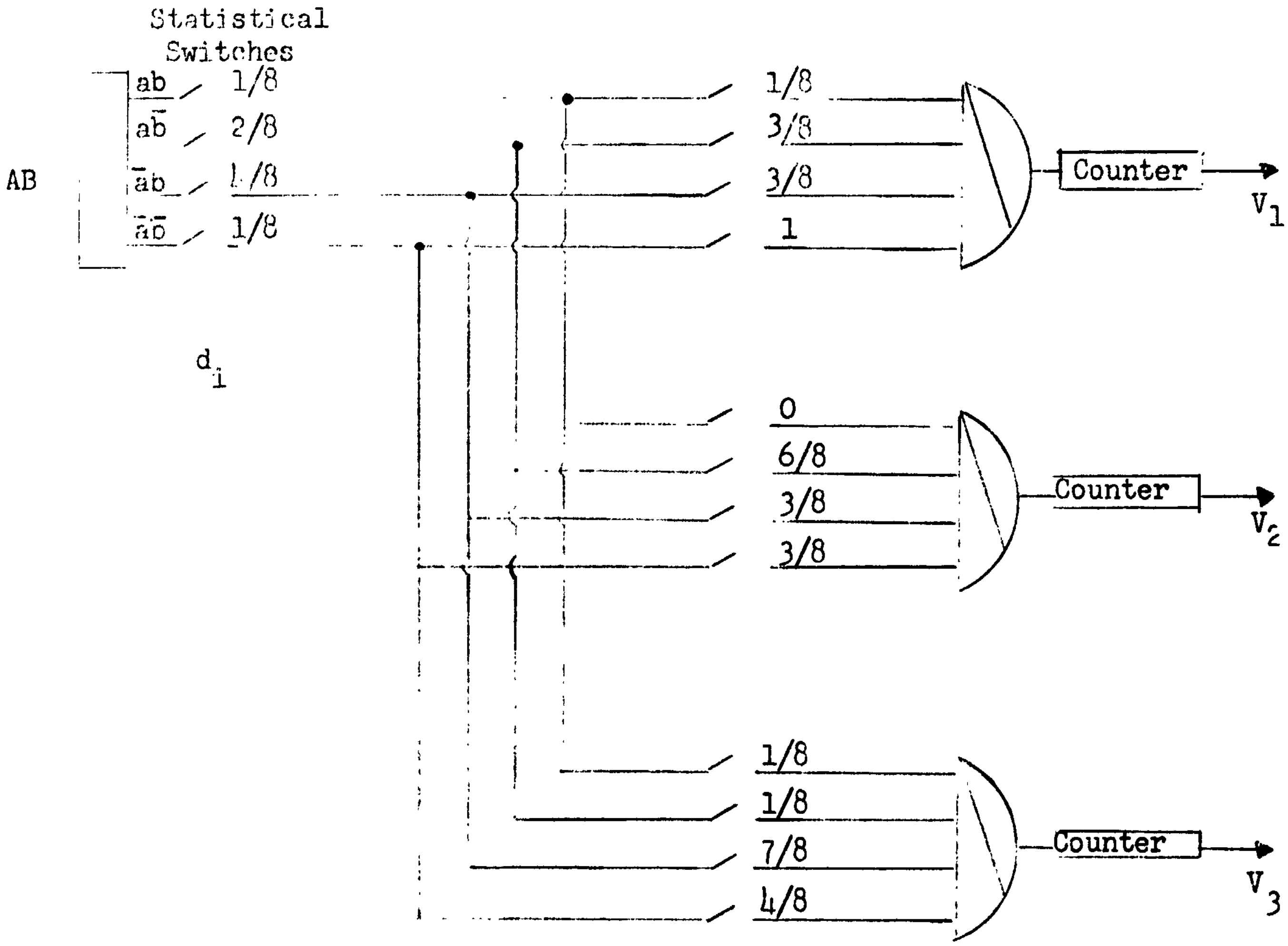
and the Bayes decision is simply the minimum (over i columns) of V_i.

The implementation of this process using Monte Carlo and statistical switches can be realized by the scheme shown in figure 3-8 which presents a simplified case of 4 states of nature with associated probabilities d_1, d_2, d_3, d_4 and three possible actions (decisions) with associated risks V_1, V_2, V_3. The operation consists of a matrix-vector multiplication by linear combination of rows:

$$\left[1/8,\ 2/8,\ 4/8,\ 1/8\right] \begin{bmatrix} 1/8 & 0 & 1/8 \\ 3/8 & 6/8 & 1/8 \\ 3/8 & 3/8 & 3/8 \\ 1 & 3/8 & 4/8 \end{bmatrix} \begin{aligned} &= 1/8\left[1/8,\ 0,\ 1/8\right] \\ &+ 2/8\ \ 3/8,\ 6/8,\ 1/8 + \ldots \end{aligned}$$

$$= \left[V_1, V_2, V_3\right]$$

The cost entries are represented by probabilities placed on the K level statistical switches in front of the OR gates (figure 3-8) where the bias is cuantized to 1 of K levels. The probability distribution vector $\vec{d}$ is represented by an extra set of statistical switches placed on the output of the minterm generator. Each minterm is activated sequentially



	V_1	V_2	V_3
$d_1 = 1/8$	1/8	0	1/8
$d_2 = 2/8$	3/8	6/8	1/8
$d_3 = 4/8$	3/8	3/8	7/8
$d_4 = 1/8$	1	3/8	4/8

Figure 3-8 Bayes Decision Computation

by an artifically generated environment (shift register with a single bit recycled) where one minterm is activated a number of times commensurate with the error variance desired on the output of each OR gate.

The final contents of each counter (V_j) then will be the sum of all products in the dot product $\vec{Q} \cdot \vec{C}$ where $\vec{C} = \left[C_{1j}, C_{2j} - - - C_{mj} \right]$ the counter containing the smallest number yielding the minimum Bayes risk.

4.0 CONCLUSIONS

4.1 Theoretical Studies

The theoretical studies have included the defining of a performance control monitoring system, development of some necessary mathematics for such systems, and the application of statistical switch techniques to performing Monte Carlo type computations.

4.2 Application Problems

Three problems have been presented, each emphasizing slightly different aspects of performance control systems. The next phase of this program will be the programming of one of the problems for simulation on a digital computer.

APPENDIX A

LEAST SQUARES PREDICTION

LEAST SQUARES PREDICTION

Let q_i be the performance measured at time t_{i+1}. It is taken as a random-variable, having mean value $E(q_i)$. Let this mean value be given by

$$E(q_i) = f(\bar{x}^i, \bar{\theta})$$

where $\bar{x}^i$ is the measurement vector at time t_i having components $x_j(t_i)$ for $j = 1, 2, \ldots, r$ and where $\bar{\theta}$ is a vector of parameters having components $\theta_j(t_i)$. Knowing this relationship enables one to establish the mean value of performance one time unit in advance of measurements $\bar{x}^i$. The problem posed is one where we are given the form of the function f. We must establish the best estimate of the parameters $\bar{\theta}$, its distributional properties, and some computing algorithm for its updating with time.

Let the measured performance over n time indices be represented by the vector

$$\bar{q} = (q_1, \ldots., q_n)$$

The problem becomes especially simple now; if we take the above referenced functional form as[13]

$$E(\bar{q}) = \bar{\theta} A^T$$

where

$$A = ((\, a_{ij} = g_j(x^i) \,)); \quad i = 1, 2, \ldots, n$$
$$j = 1, 2, \ldots, r$$

with $g_j(x^i)$ being independent functions of the measurement vector $\bar{x}$ at time-index i, and where the parameter vector is of the form

$$\bar{\theta} = (\theta_1, \theta_2, \ldots, \theta_r).$$

That is, the performance predicted for time index i+1 is given by

$$E(q_i) = f(\bar{x}^i, \bar{\theta}) = \sum_{j=1}^{r} a_{ij} \theta_j$$

$$= a_{i1} \theta_1 + a_{i2} \theta_2 + \ldots + a_{ir} \theta_r$$

$$= g_1(\bar{x}^i)\theta_1 + g_2(\bar{x}^i)\theta_2 + \ldots + g_n(\bar{x}^i)\theta_r$$

for i = 1, 2, ...n.

Select the "best" estimate of the parameter$\bar{\theta}$, $\hat{\theta}$, as that which minimizes the sum of the squares of the deviation of $E(q)$ from the actual measurements q.

That is, let

$$R^2 = \left[\bar{q} - E(\bar{q}) \right] \left[\bar{q} - E(\bar{q}) \right]^T$$

$$= \left[\bar{q} - \bar{\theta}A^T \right] \left[\bar{q} - \bar{\theta}A^T \right]^T$$

To minimize R^2, let the rank of A be r, whereby it can be shown in a straight-forward manner (setting its derivative with respect to $\bar{q}$ equal to the vector $\bar{O}$) that one must select

$$\hat{\theta} = \bar{q}A \; (A^T A)^{-1}$$

(Note that matrix A is not a square matrix, and that A^TA is a non-singular symmetric r x r matrix).

In this case, one obtains

$$R^2 = \left[\bar{q} - \hat{\theta} A^T\right] \left[\bar{q} - \hat{\theta} A^T\right]^T$$

$$= \bar{q} \left[I - A(A^T A)^{-1} A^T\right] \bar{q}^T$$

A theorem by Markov states that if we take the components of q as normally distributed with common variance σ^2 (a restriction that's convenient rather than necessary,

then

(1) $\hat{\theta} \cap N \left[\bar{\theta}, \;\; ^2 \; (A^T A)^{-1}\right]$

(2) $\frac{R^2}{\sigma^2} \cap \chi^2_{n-r}$

(3) $\hat{\theta}$ and $\frac{R^2}{\sigma^2}$ are independent,

where $\cap$ is used to denote "distributed as" and N $[a, B]$ used to denote "normal with mean a and variance-covariance B". This result is of interest to us here in that one obtains the distributional properties of the prediction, required for decision theoretic considerations.

As an illustration let $f(\bar{x},\bar{\theta})$ be some arbitrary function of, say, three variables. Let these be the three <u>measurements</u>; x_1, x_2, and x_3. The best second-order fit of $E(\bar{q}_1)$ is of the form

$$E(q_i) = x_1^{(i)} \hat{\theta}_1 + x_2^{(i)} \hat{\theta}_2 + \ldots\ldots + x_2^{(i)} \; x_3^{(i)} \; \hat{\theta}_{23}$$

where $\hat{\theta} \equiv (\hat{\theta}_1, \hat{\theta}_2, \ldots, \hat{\theta}_{23})$

is evaluated as above using the matrix A given by

$$A = \begin{bmatrix} x_1^{(1)}, & x_2^{(1)}, & x_3^{(1)}, & x_1^{(1)\,2}, & x_2^{(1)\,2}, & x_3^{(1)\,2}, & x_1^{(1)} x_2^{(1)}, & x_1^{(1)} x_3^{(1)}, & x_2^{(1)} x_3^{(1)} \\ \\ x_1^{(n)}, & x_2^{(n)} & x_3^{(n)}, & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots \end{bmatrix}$$

The previously described procedure now yields the least squares fit of the specified form.

With n > r collections of data (consisting of the n rows of the matrix A) one can readily establish $\hat{\theta}$. If the distribution of $\bar{q}$ is known, one can also establish the distribution of $\hat{\theta}$. For the time being, however, we'll consider $\bar{q}$ as normally distributed. Consequently, as n gets larger the elements of the variance-covariance matrix associated with θ decreases approximately as $\frac{1}{n}$. Whereas this property is desired when the process considered is stationary, it may not be desired in the adaptive prediction techniques considered here.

Consider first the case where N sets of data are to be considered in conjunction with weighting factors which depend only on their age. The solution is obtained by using a modified R^2, Q^2, given by

$$Q^2 = \left[\bar{q} - \theta A^T\right] \begin{bmatrix} w_1 & 0 & 0 & \cdot & \cdot & \cdot \\ 0 & w_2 & 0 & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & & & \\ \cdot & \cdot & & \cdot & & \\ \cdot & \cdot & & & \cdot & \\ \cdot & \cdot & & & & w_n \end{bmatrix} \left[\bar{q} - \theta A^T\right]^T$$

Where the w_j are positive weighting factors. By denoting the weighting matrix S, we obtain

$$Q^2 = qWq^T - qWA\theta^T - \theta A^T Wq + \theta A^T WA\theta^T$$

wherein

$$\left.\frac{\partial Q^2}{\partial \theta}\right|_{\theta = \hat{\theta}} = -2qWA + 2OA^TWA = 0$$

Hence,

$$\hat{\theta} = \bar{q}\ (WA)\ (A^TWA)^{-1}$$

where $(A^TW\ A)^{-1}$ exists whenever $(A^T\ A)^{-1}$ exists, (i.e., whenever A is of rank r, and $\theta = \theta_1, \ldots, \theta_r$). One can again obtain the distribution in $\hat{\theta}$ by knowing the distribution of $\bar{q}$.

We shall now consider the computational aspects of the above. Designate the performance measurements by the vector

$$\bar{q}_o = (q_1, q_2, \ldots, q_n).$$

Let the parameters that have been estimated after the above n - measurements, be designated $\hat{\theta}_n$. Let the system measurement matrix be arranged in the (n x r) matrix, A_n. We've indicated that the simple unweighted case gives rise to the solution

$$\hat{\theta}_n = \bar{q}_n A_n (A_n^T A_n)^{-1}$$

If the additional data obtained during the (n+1) $\underline{\text{st}}$ interval is designed by the (vector) row-matrix $B_{n+1} = a_{n+1,1}, a_{n+1,2}, \cdots, a_{n+1,r})$, then the updated "best" estimate of parameter θ is representable in terms of the partitioned matrices given in the equation.

$$\theta_{n+1} = (q_o, q_{n+1}) \begin{bmatrix} A_n \\ B_{n+1} \end{bmatrix} \left\{ \begin{bmatrix} A_n^T & B_{n+1}^T \end{bmatrix} \begin{bmatrix} A_n \\ B_{n+1} \end{bmatrix} \right\}^{-1}$$

wherein

$$\hat{\theta}_{n+1} = \left[\bar{q}_o A_n + q_{n+1} B_{n+1} \right] \left[A_n^T A_n + B_{n+1}^T B_{n+1} \right]^{-1}.$$

The total system - memory required for this updating process resides in the (1 x r) vector $\left[\bar{q}_o A_n \right]$ and the (r x r) - matrix, $A_n^T A_n$.

For the weighted case, we established that for n sets of data,

$$\hat{\theta}_n = \bar{q}_o W_n A_n (A_n W_n A_n)^{-1}$$

where

$$W_n = \begin{bmatrix} w_1 & 0 & 0 & \cdots & \cdots \\ 0 & w_2 & 0 & \cdots & \cdots \\ \cdot & \cdot & \cdot & & \\ \cdot & \cdot & & \cdot & \\ \cdot & \cdot & & & \ddots \\ & & \cdots & \cdots & w_n \end{bmatrix}$$

This implies that the weight w_j is given to the square of the $j^{\underline{th}}$ deviation, relative to the minimization procedure. It is simplest to assume that the relative importance of one sample to another sample remains fixed once it is established. That is, cnce the weight w_j is established, the ratio w_j/w_k for all $k \leq j$ remains fixed (independent of assignments of weights w_i for $i > j$). Under this assumption, we see that for time index n+1 we can write

$$\hat{\theta}_{n+1} = (q_o, q_{n+1}) \begin{bmatrix} W_n & 0 \\ 0 & w_{n+1} \end{bmatrix} \begin{bmatrix} A_n \\ B_{n+1} \end{bmatrix} \left[(A_n^T, B_{n+1}^T) \begin{bmatrix} W_n & 0 \\ 0 & w_{n+1} \end{bmatrix} \begin{bmatrix} A_n \\ B_{n+1} \end{bmatrix} \right]^{-1}$$

wherein our weighted updating procedure would compute

$$\hat{\theta}_{n+1} = \left[q_o W_n A_n + w_{n+1} q_{n+1} B_{n+1} \right] \left[A_n^T W_n A_n + w_{n+1} B_{n+1}^T B_{n+1} \right]^{-1}$$

Noting that the up-dated parameters are represented as

$$\bar{q}_o = (\bar{q}_o, q_{n+1})$$

$$A_{n+1} = \begin{bmatrix} A_n \\ B_{n+1} \end{bmatrix}$$

$$W_{n+1} = \begin{bmatrix} W_n & 0 \\ 0 & w_{n+1} \end{bmatrix}$$

we see that $q_o W_{n+1} A_{n+1} = q_o W_n A_n + w_{n+1} q_{n+1} B_{n+1}$

and

$$A_{n+1}^T W_{n+1} A_{n+1} = A_n^T W_n A_n + w_{n+1} B_{n+1}^T B_{n+1}$$

These parameters are then used in the succeeding cycles of computations, yielding θ_{n+m}, for all m.

The above scheme can be readily modified in order to perform the computation of predicted performance on the basis of previous data <u>without</u> updating. This would be useful for establishing control modifications of the man-machine process.

APPENDIX B

THEORY OF ORGANIZING TRAINABLE LOGICAL NETWORKS

MATHEMATICAL REPRESENTATION OF A TRAINABLE LOGICAL NETWORK

Consider a binary logical network with n-input variables and n-output variables.

Let: $v_1\ v_2\ v_3\ \dots\ v_n$ be input variables

$w_1\ w_2\ w_3\ \dots\ w_n$ be output variables

$x_1\ x_2\ x_3\ \dots\ x_{2^n}$ be combinations of input variables

$y_1\ y_2\ y_3\ \dots\ y_{2^n}$ be combinations of output variables

If we associate with each of the input combinations a unit m-tuple (where $m = 2^n$), then any subset of the entire set of input combinations can be represented by an m-tuple which is a linear combination of the unit m-tuples. That is, if the unit m-tuples are

$e_1 = 1\ 0\ 0\ 0\ \dots\ 0$

$e_2 = 0\ 1\ 0\ 0\ \dots\ 0$

$e_3 = 0\ 0\ 1\ 0\ \dots\ 0$

.

.

.

$e_{2^n} = 0\ 0\ 0\ 0\ \dots\ 1$

then any class of input combinations can be represented by an m-tuple,

$$X = \sum_{i=1}^{2^n} x_i' e_i$$

where $x_i' = 1$ if the combination x_i is in the desired input set. Otherwise $x_i' = 0$.

Similarly with the outputs

$$Y = \sum_{i=1}^{2^n} y_i' e_i$$

where $y_i' = 1$ if y_i is to be specified in the output set. Otherwise, $y_i' = 0$.

Now let switches be labeled S_{ij} so that S_{ij} is the connection from input combination x_i to output variable w_j. For an n-input, n-output network there will be, in general, $n2^n$ switches ($i = 1, 2, \ldots, 2^n$ and $j = 1, 2, \ldots, n$). If to each of the switches is associated a value a_{ij} such that $a_{ij} = 1$ when switches S_{ij} are closed and $a_{ij} = 0$ when S_{ij} is open, then input combination x_k appears at the output as $y = e_k A$ where e_k is the m-tuple corresponding to input combination x_k and $A = (a_{ij})$.

The admissible set of outputs for a given input set X will be

$$Y_x = X_d A$$

where X_d is the m-tuple X written in diagonal form.

The admissible set of outputs for all possible inputs will be

$$y_o = \begin{bmatrix} e_1 \\ e_2 \\ \cdot \\ \cdot \\ \cdot \\ e_{2^n} \end{bmatrix} \qquad A = \begin{bmatrix} 1\,0\,0\,0 \ldots 0 \\ 0\,1\,0\,0 \ldots 0 \\ \cdot \\ \cdot \\ \cdot \\ 0\,0\,0\,0 \ldots 1 \end{bmatrix} \qquad A = IA = A$$

where the row j of A is the output corresponding to input x_j. The output set is again representable as a vector

$$Y = \sum_{i=1}^{2n} y_i' e_i \qquad \begin{aligned} y_i' &= 1 \; y_i \varepsilon A \\ y_i' &= 0 \; y_i \not\varepsilon A \end{aligned}$$

Example:

transform $\underline{x}$ to $\underline{y}$

0 1 0 → 1 0 1

1 0 1 → 0 0 1

1 1 0 → 0 1 0

all others → 0 0 0

Let $x_i = v_1\ v_2\ v_3$ for all possible values of the input variables

i.e., $x_1 = 0\ 0\ 0$ and $e_1 = 1\ 0\ 0\ 0\ 0\ 0\ 0\ 0$

$x_2 = 0\ 0\ 1$ $e_2 = 0\ 1\ 0\ 0\ 0\ 0\ 0\ 0$

$x_3 = 0\ 1\ 0$ $e_3 = 0\ 0\ 1\ 0\ 0\ 0\ 0\ 0$

$\vdots$ $\vdots$

$x_8 = 1\ 1\ 1$ $e_8 = 0\ 0\ 0\ 0\ 0\ 0\ 0\ 1$

The transformation matrix can be written down immediately. Each output combination must appear as at least one row of a_{ij}. Specifically, row i is the output associated with input x_i, so

$$A = (a_{ij}) = \begin{matrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 1 & 0 & 1 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \\ 0 & 1 & 1 \\ 0 & 0 & 0 \end{matrix}$$

$a_{ij} = 1$ indicates that switch S_{ij} is closed.

If, instead of associating values $a = (0, 1)$ with switches, we associate a range of values $0 \leq a'_{ij} \leq 1$, then a'_{ij} becomes the probability of the switch being closed; i.e.,

$$a'_{ij} = \text{Prob } (a_{ij} = 1)$$

and the n-tuple

$$a'_i = \sum_j^n a'_{ij} e_j$$

defines the statistical states with respect to input combination x_i. The statistical state of the total system is representable by the matrix A'.

$$A' = (a'_{ij})$$

A change in the statistical state of the system $A'_1 \rightarrow A'_2$ can be effected by changing values of a'_{ij}. If values are changed on successive inputs, then the transformation

$$A'_1 \rightarrow A'_2 = A'_1 + \Delta A'_1$$

will involve at most only some a'_k ; i.e.,

$$\Delta A' = (a'_k: q_k \rightarrow q_r \; \Delta a'_k)$$

where q_k and q_r are values of a'_k, and

$$P_{kr} = \text{Prob } (\Delta a'_k = q_k \rightarrow q_r)$$

where P_{kr} is the probability of transition from statistical state q_k to q_r. If the same rule is used for determining a'_k on successive trials, then the transition probabilities are functions of the states and the rule only; i.e.,

$$P_{rk} = f(q_k, q_r)$$

The probabilities of P_{kr} then define a Markov process, since they are functions of the states and are independent of how the systems arrived at state q_k.

MATHEMATICAL INTERPRETATION OF THE GOAL FUNCTION

Previously, when the transition probabilities for the statistical states of the machine were being developed, it was assumed that a rule for determining new statistical states could be found. The new statistical state was determined to be the old statistical state plus some incremental change and the probability of that change occurring was the transition probability.

Recall that if the present statistical state is

$$a_i^{'}(t) = q_k$$

and $\quad a_i^{'}(t+1) = q_r$

then $\quad a_i^{'}(t+1) = a_i^{'}(t) + \Delta\, a_i^{'}(t)$

or $\quad \Delta a_i^{'}(t) = a_i^{'}(t+1) - a_i^{'}(t)$

so that Prob $\Delta a_i^{'}(t) =$ Prob $(q_k \rightarrow q_r)$

In somewhat greater detail

$$\Delta a_i^{'} = \left[a_{i1}^{'}(t+1) + a_{i2}^{'}(t+1) \ldots\right] - \left[a_{i1}^{'}(t) + a_{i2}^{'}(t) \ldots\right]$$

$$= \left[a_{i1}^{'}(t+1) - a_{i1}^{'}(t)\right] \left[a_{i2}^{'}(t+2) - a_{i2}^{'}(t) \quad \ldots\right]$$

$$= \Delta\, a_{i1}^{'}(t)\ \Delta a_{i2}^{'}(t) \ldots \Delta\, a_{im}^{'}(t)$$

There is no reason for us to expect that the increments comprising $a_i^{'}$ need all be equal. In fact, by making the increments equal, we cannot get to a majority of the statistical states. However, in the absence of very specific information about the desired statistical state, it is difficult to see why any one component of $\Delta a_i'$ should receive greater changes than another. Intuitively, changes $\Delta\ a_i'$ depend upon some measure of the machine's outputs. The measure m(y), in the simplest case, is m(y) = 1 if the machine gives the desired output and m(y) = 0 for an undesired output.

Clearly, a range $0 \leq m(y) \leq 1$ is possible which could be a measure on the probability that y(t) is a desired output. Further, the measure need not be on a single output combination. One might possibly use a measure m(y') over an output sequence. However, most of our work to date has used the following simple goal function for determining $\Delta a_i'$.

Let R_c be a class of desired outputs. Then Δa_i has been the following function F(m(y)), a(t) of the machines outputs and logical states.

$$\Delta\ a_i' = F\ \left[m(y),\ a(t)\right]$$

$$= C\ \left[2m(y) - 1\right]\ \left[a_i(t) - a(t)\right]$$

where c < 1. Thus, all components $\Delta a_{ij}'$ of $\Delta a_i'$ are of equal magnitude (c) but may vary in sign. A single component of $\Delta a_i'$ is just:

$$\Delta\ a = C\ \left[2m(y) - 1\right]\ \left[a_{ij} - a_{ij}\right]$$

where: $m(y) = 1 \qquad y(t) \varepsilon R_c$

$m(y) = 0 \qquad y(t) \not\varepsilon R_c$

$c < 1$

To find the new statistical state we observe:

$$A'(t+1) = a'(t) + \Delta\ a'(t)$$

The question naturally arises as to what happens when a component of a_i' is equal to 1 and $\Delta\ a_i'(t)$ is not 0. It may be recalled that, in derivation of statistical states, a_{ij}' was the probability of closure of switch S_{ij}.

The problem is only momentarily embarrassing and can be surmounted either by redefining the statistical states so that their components no longer represent probabilities or by allowing $\Delta a'_{ij}(t)$ only to take on values such that $0 \leq a' (t+1) \leq 1$. The latter is chosen to preserve the statistical state concept.

In order that there be no loss of information, the goal function remembers the total $\sum_t \Delta a'_{ij}(t)$ and adds increments $\Delta a'_{ij}$ to a'_i only when $0 \leq \sum_t \Delta a'_{ij}(t) \leq 1$. The initial statistical state may be defined as

$a'_{ij}(1) = \Delta a'_{ij} (0)$.

In practice, the memory and the statistics of the switch are usually closely associated and the probability of switch closure is given by:

$$\begin{aligned} a'(j) &= 0 && j < r \\ a'(j) &= \frac{j-r}{N-2r} && r \leq j \leq (N - r) \\ a'(j) &= 1 && (N - r) < j \end{aligned}$$

where: $N_o = N\, a'(0)$

r = goal function memory capacity

$j = N_o + \sum_t (2\, m(y) - 1)\, (a - \bar{a})$

N = arbitrary number of switch levels plus associated memory

THE TRAINING PROCESS

State Representation

As was indicated earlier, the behavior of a machine at any time can be characterized by its statistical states and goal function. To see this more explicitly, let us examine the switches associated with a particular input combination x_i. The switch involved would be $S_{i1}\ S_{i2} \dots S_{in}$. The respective probability of closure on each switch would be $a'_{i1}\ a'_{i2} \dots a'_{in}$. This combination of values is the statistical state with respect to the input combination x_i. Frequently, it will be referred to as the statistical state.

It should be observed that a'_i is not a number but rather is a function whose values define the statistical states.

$$a'_i = a'_{i1}\ a'_{i2} \dots a'_{in}$$

If each a'_{ij} can take on r values then a'_i can take on $(r)^n$ values.

Transition Probabilities

Letting $a'(t) = q_j$ be the value of a'_i at trial t, then the transition probability to some other state q_k on the next trail becomes

$$p_{jk} = \text{Prob}\ \left[a'_i\ (t+1) = q_k\ \Big|\ a'_i(t) = q_j\right]$$

The transition probability p_{jk} is said to characterize a Markov process if, for any statement g whose validity is determined by $a'(t_1)$ where $t_1 \leq t$, the transition probability remains unchanged; i.e.,

$$p_{jk} = \text{Prob}\ \left[a'(t+i) = q_k \mid a'(t) = q_k \cap g\right]$$

This assures that the transition probability of going from state q_j to state q_k is independent of the sequence of states leading to q_j. The transition probability p_{jk} tacitly assumes $a_i'(t) = q_j$. Without this assumption, the transition probabilities will generally appear time-dependent. If without the assumption $a'(t) = q_j$ the transition probability $p_{jk}(t+1)$ is independent of t, then the process is said to be stationary and the states are sometimes referred to as ergodic states.

Transition Matrix

Previously, we saw the development of transition probabilities p_{jk} from a statistical state q_j to a state q_k. The total picture can sometimes be more easily visualized by means of a matrix of transition probabilities.

$$P = (p_{jk})$$

In the matrix P, the kth element of row j is the probability of transferring from state q_j to state q_k

$$P = \begin{array}{c|cccc|}
 & q_1 & q_2 & q_k & q_n \\
q_1 & p_{11} & p_{12} & \cdots & p_{1n} \\
q_2 & & & & \\
\cdot & & & & \\
\cdot & & & & \\
q_j & \cdots & & p_{jk} & p_{jn} \\
\cdot & & & & \\
q_n & p_{n1} & & \cdots & p_{nn}
\end{array}$$

The notation of the transition matrix is useful because it allows us to calculate various parameters which describe the process and is helpful in classifying various types of processes.

For example, to calculate the transfer probability from state q_j the state q_k as a function of the number of trials, we observe the following:

Let $p_{jk}^{(m)}$ be the probability of finding the system in state q_k after m-trials

(1) $p_{jk}^{(1)} = p_{jk}$ First trial

(2) $p_{jk}^{(2)} = \sum_r p_{jr} \, p_{rk}$ Second trial

(3) $p_{jk}^{(m)} = \sum_r p_{jr} \; {}^{(m-1)}p_{rk}$ In general for m-trials

(4) By further induction

$$p_{jk}^{(m+n)} = \sum_r p_{jr}^{(m)} \, p_{rk}^{(n)}$$

In matrix notation

$$\left[p_{jk}^{(m)}\right] = \left[p_{jr}^{(m-1)}\right] \left[p_{rk}\right]$$

$$= p^m = p^{m-1} \, p$$

$$= \left\{\text{Prob} \left[a(m) = q_k \;\; a(o) = q_j\right]\right\}$$

We see then that the powers of the original transition matrix can give us the probability of being in new states after several trials.

The study of Markov processes is a rather broad subject and no attempt will be maae here to explore the subject except as it relates directly to the organization of a machine.

Classification of Markov Chains

In classifying Markov chains, we visualize a process which moves from state to state. Classes of chains are obtained by equivalences derived by restrictions imposed upon the movement from state to state.

Consider the situation where there exists two sets of states such that, with the first set, the process can move to the second set, but, once in the second set, the process can no longer get back to the first set of states. States belonging to the first set of states are called transient states. States belonging to the second set of states form an absorbing chain. If an absorbing chain consists of one state, then it is referred to as an absorbing state. An absorbing state then is immediately recognizable in a transition matrix, since it will have transition probabilities of 0 to all states except itself; i.e.,

$$p_{ij} = 0 \qquad i \neq j$$
$$p_{ij} = 1 \qquad i = j$$

The behavior of a Markov process can, to a large degree, be characterized by the type of sets which make up the chains. Chains which have no transient can be broken down into two types. They are <u>regular</u> and <u>cyclic</u> chains. In <u>regular</u> chains the transition matrix will have no 0 entries for sufficiently high powers. The process can be in any state after a large number of transitions. In <u>cyclic</u> chains there will always be some 0 entries in all powers of the transition matrix. The process is predictable to subsets of the chain and will move in a more orderly fashion from state to state, eventually returning to its starting state.

In chains with transient sets, we observe that it is implicit that such chains also contain absorbing sets. The number of distinct absorbing sets and their size serves to classify the process in this case. Thus, a chain with transient sets may contain cyclic, regular, or unit absorbing sets.

Occasionally, it is convenient to classify states within chains into two broad classes. These are transient states and nontransient states (ergodic).

We will examine more closely the behavior of processes as they move through the chains. First, we will consider chains with transient sets and show that the process inevitably must leave the transient set. Second, we will consider a chain with no transient sets and determine a time-invariant probability distribution of states.

The Behavior of a Process in Chains

Intuitively, it is quite clear that, if a process moving through a chain can leave a particular set of states but cannot enter into the set once it has left, eventually the process will not be found in the transient set. To see this analytically, we recall that the probability of finding the process in state q_k, m-trials after starting in state q_j is:

$$\overset{(m)}{p_{jk}} = \sum_r \overset{m-1}{p_{jr}} \overset{1}{p_{rk}}$$

It is just the probability of transferring to state q_r in m-1 trials and then to q_k on the next trial summed over all possible q_r.

Letting q_j be the starting state in a transient set (T) and q_k be any other state in (T), we see that

$$\sum_k p_{jk}^{(m)} < 1$$

since it is possible to leave the set. It follows immediately that

$$\lim_{m \to \infty} \sum_{k} p_{jk}^{(m)} \to 0$$

In matrix form

$$p^{m} \to 0 \text{ as } m \to \infty$$

The mean number of trails for which the process remains in the transient set is all q_k in T, given that the starting point in state q_j is

$$M_j = \sum_{m=o}^{\infty} (1) \quad \Big(\sum_{k} p_{jk}^{(m)}\Big)$$

and the mean time in any particular state q_k is

$$M_{jk} \sum_{m=o} (1) \quad (p_{jk}^{m})$$

since each transition contributes one trial. The matrix which gives the mean time remaining in the transient set starting from any transient state becomes

$$M = \sum_{m=o} p^{m}$$

This quantity is somewhat easier to compute in an alternate form

$$M = (I - P)^{-1}$$

where P is the transition matrix for the transient states. To see that these forms are equivalent, observe

$$(I - P)\,(I + P + P^{2} \ldots P^{m}) = I - p^{m+1}$$

For $m \to \infty$, we have shown $P^{m+1} \to 0$ thus,

$$(I - P)(I + P + P^2 \dots P^m) = I$$

$$(I - P)(\sum_{m=o}^{\infty} P^m) = I$$

Since this matrix product is nonsingular, we may write

$$\sum_{m=o}^{\infty} P^m = (I - P)^{-1}$$

If a chain has no transient sets, then it is possible for the process to get to any state of the chain. The fraction of time that the process spends in each state will then form a probability density function that the process will be found in a given state. Previously, we have shown that, if it is possible for a system to leave a state, eventually it will. To observe the behavior of such a system, we assume that the time average state configuration of one system is equal to the instantaneous ensemble average of many similar systems.

Looking at an ensemble of systems, we see that, if there exists an initial distribution $F(q_j)$ of systems in each state $F = f_1 f_2 \dots f_j \dots f_m$ where f_j is the fraction of systems in state q_j, such that the distribution is time-invariant, then the process is stationary. For the fraction of systems in each state to remain constant, the number of systems leaving a given state must be equal to the number of systems entering that state. The fraction of systems leaving state q_j is simply the product of the fraction of systems in that state with the transition probabilities to all other states. Thus;

Fraction leaving $q_j = f_j\, p_{j1} + f_j\, p_{j2} + \dots + f_j\, p_{jm}$

$$= f_j \sum_k p_{jk}$$

$$= f_j$$

Since

$$\sum_k p_{jk} = 1$$

and

Fraction entering $q_j = f_1\, p_{1j} + f_j\, p_{2j} + \dots + f_m\, p_{mj}$

$$= \sum_i f_i\, p_{ij}$$

So

$$f_j = \sum_i f_i\, p_{ij}$$

in matrix form for all states

$$F = F\,P$$

taking transposes of both sides

$$F^T = p^T\, F^T$$

or

$$(P^T - I)\, F^T = 0$$

which is a homogeneous set of equations in $F(F = f_1\, f_2 \dots f_m)$ and has a solution only if the determinant of the coefficients vanishes; i.e.,

$$\left| p^T - I \right| = 0$$

This criterion establishes a necessary condition for the distribution F to remain time-invariant. Clearly, when the solutions to the homogeneous set of equations has $f_j \neq 0$ for all j, there are no transient sets; and, when $f_j = 0$ for at least some j, there exists at least one transient set.

APPENDIX C

TRUNCATED SEQUENTIAL DECISIONS

Bayes Truncated Sequential Decision Solution

Let the loss function associated with one of a finite set of possible states of nature, $\omega \in \Omega$, and one of a finite set of possible actions, $a \in A$, be denoted $L(a,\omega)$. If one selects that action which minimizes the expected loss after k - experiments have been performed, then this expected loss can be shown(8) to be given by

$$U_k = \sum_{i=1}^{k} C_i(x_1,\ldots,x_i) + \operatorname*{Min}_{a \in A} E_{k\xi}\left[L(a,\omega)\right]$$

where C_i is the cost of the $i^{\underline{th}}$ experiment which yielded measurement x_i. (If A is not finite, one simply replaces "minimum" with "infimum" over A.) In this expression, $\xi(\omega)$ is the distribution over the states of nature and

$$E_{k\xi}\left[L(a,\omega)\right] \equiv \frac{\sum_{F(\vec{x})}\sum_{\omega \in \Omega} L(a,\omega)\, p(\vec{x}\mid x_1, x_2, \ldots, x_k, \omega)\,\xi(\omega)}{\sum_{F(\vec{x})}\sum_{\omega \in \Omega} p(\vec{x}\mid x_1, \ldots, x_k, \omega)\;\xi(\omega)}$$

where $F(x)$ is the set of all x whose first k-coordinates are $x_1, \ldots, x_k$.

To establish whether to continue experimenting or to make a decision after a given experiment, one compares the expected loss associated with each of these possibilities. The lesser of these two expected losses (after k-experiments have been performed) is given by α_k. After the complete set of N-experiments have been run, the minimum loss would be

$$\alpha_N = U_N = \sum_{i=1}^{N} C_i(x_1, x_2, \ldots, x_i) + \operatorname*{Min}_{a \in A} E_{N\xi}\left[L(a,\omega)\right]$$

Hence, after N-1 experiments it would be

$$\alpha_{N-1} = \text{Smaller of the two numbers} \left\{ \begin{matrix} U_{N-1} \\ E_{N-1,\xi}\left[\alpha_N\right] \end{matrix} \right\}$$

Continuing in this way, one obtains the complete set of minimum risks at each stage of experimentation to be

$$\alpha_N = U_N$$

$$\alpha_{N-1} = \text{Smaller}\left\{ U_{N-1},\ E_{N-1,\xi}\left[\alpha_N\right] \right\}$$

$$\vdots$$

$$\alpha_j = \text{Smaller}\left\{ U_j,\ E_{j\,\xi}\left[\alpha_{j+1}\right] \right\}$$

$$\vdots$$

$$\alpha_0 = \text{Smaller}\left\{ U_0,\ E_{\xi}\left[\alpha_1\right] \right\}$$

where U_0 is the expected loss associated with making a decision without experimentation, given by

$$U_0 = \min_{a \in A} \sum_{\omega \in \Omega} L(a,\omega)\,\xi(\omega)$$

The Bayes optimal procedure requires computation of α_N, α_{N-1}, ..., α_0, in this order. At each stage of experimentation, say j, one makes a decision if

$$\alpha_j = U_j$$

Otherwise, one continues with experiment j + 1. This is represented schematically by the tree-structure shown below in figure C-1.

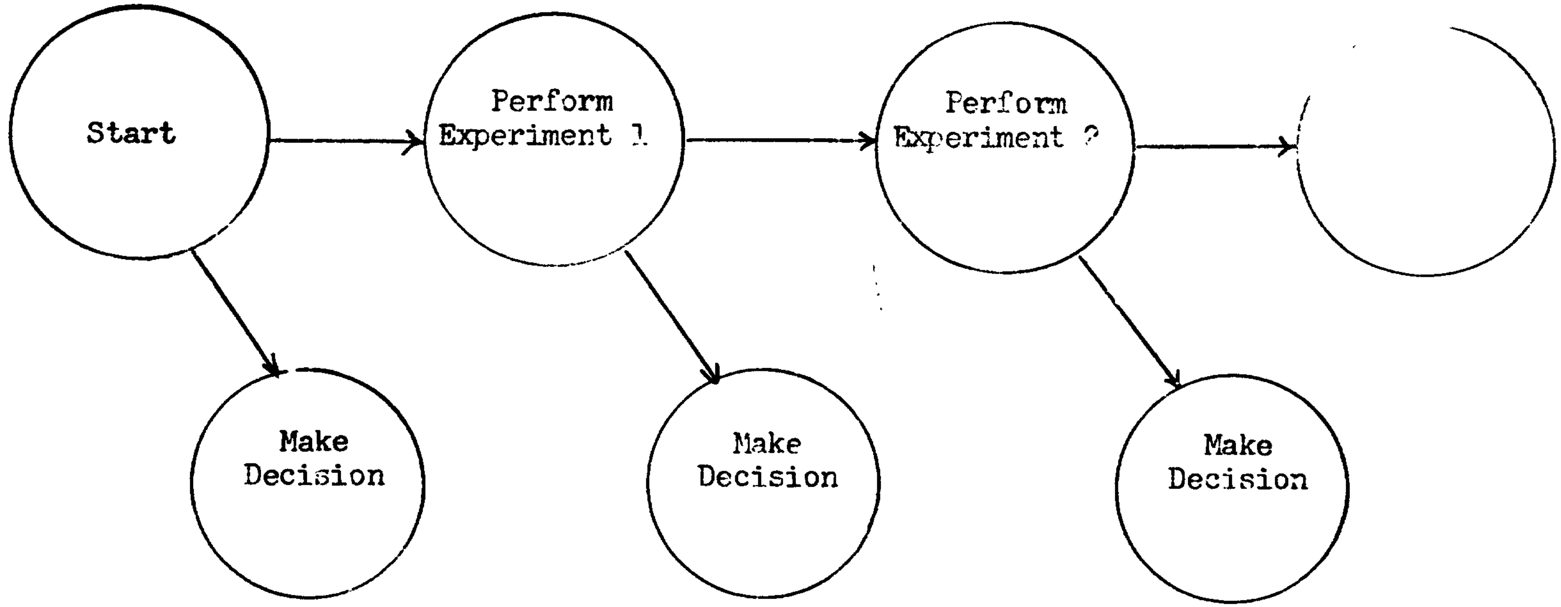


Figure C-1 Decision Tree - Sequential Theory with Fixed Ordered Experiments

Truncated Sequential Decisions with Determination of the Order of Experimentation

Because notation becomes cumbersome, the procedure is described for the case of three experiments e_1, e_2, e_3. Generalization to N-experiments follows directly, as does a rigorous proof. The description given below, however, makes for greater clarity.

Basically, the process is like that described previously. The loss matrix will be given by $((L(a,\omega)))$. Let the states of nature, $\omega \in \Omega$ and the set of possible actions, $a \in A$, be finite. Let the set of observations resulting from the experiments be denoted by the vector $\vec{x} = (x_1, x_2, x_3)$, where coordinate x_i corresponds to experiment e_i. The minimum expected loss without experimentation is then given by[12]

$$U_{\emptyset} = \operatorname*{Min}_{a \in A} E_{\xi}\left[L(a,\omega)\right]$$

where $\xi(\omega)$ is the a priori distribution over the states of nature and

$$E_{\xi}\left[L(a,\omega)\right] = \sum_{\omega \in \Omega} L(a,\omega)\,\xi(\omega)$$

After experiment e_i has been performed, yielding the result $x_i = x_i^0$, the minimum expected loss associated with making a decision is given by

$$U_i = \min_{a \in A} E_{\xi^i}\left[L(a,\omega)\right] + C_i(x_i^0)$$

where $C_i\,(x_i^0)$ is the cost of performing experiment e_i and having the results be $x_i = x_i^0$, and where

$$E_{\xi^i}\left[L(a,\omega)\right] = \frac{\sum\limits_{F(\vec{x}|x_i^0)}\sum\limits_{\omega\in\Omega} L(a,\omega)\,p(\vec{x}|x_i = x_i^0,\omega)\,\xi(\omega)}{\sum\limits_{F(\vec{x}|x_i^0)}\sum\limits_{\omega\in\Omega} p(\vec{x}|x_i = x_i^0,\omega)\,\xi(\omega)}$$

with the set $F\ (\vec{x}|x_i^0)$ used to indicate summation is taken over the set of all possible $\vec{x}$ whose $i^{\underline{th}}$ coordinate is $x_i = x_i^0$. (The symbol p is used generically to represent "probability".) In a like manner, the minimum expected loss associated with making a decision after performing experiment e_i and then performing e_j, obtaining results x_i and x_j, is given by

$$U_{ij} = \min_{a \in A} E_{\xi ij}\left[L(a,\omega)\right] + C_i(x_i^0) + C_{ij}\ (x_i^0, x_j^0)$$

where $C_{ij}\ (x_i^0, x_j^0)$ is the cost of experiment e_j after e_i has been performed and where

$$E_{\xi ij}\left[L(a,\omega)\right] = \frac{\displaystyle\sum_{F(\vec{x}|x_i^0,x_j^0)} \sum_{\omega \in \Omega} L(a,\omega)p(\vec{x}|x_i^0,x_j^0,\omega)\xi(\omega)}{\displaystyle\sum_{F(\vec{x}|x_i^0,x_j^0)} \sum_{\omega \in \Omega} p(\vec{x}|x_i^0,x_j^0,\omega)\ \xi(\omega)}$$

For the general case considered here, one should note that

$$p(\vec{x}\ |\ x_i^0,\ x_j^0,\ \omega) \neq p\ (\vec{x}\ |\ x_j^0,\ x_i^0,\omega)$$

That is, the order in which experiments are performed can be expected to be different if, for example, the experiments alter the state of the system considered. However, this procedure is actually required (in general) whenever these probabilities are not independent.

The procedure at each stage in experimentation is to compare the expected loss associated with stopping experimentation and the expected loss for the various continuations. As before, the various expected losses are established by first computing

$$\begin{array}{lll}
\alpha_{123} = U_{123} & \qquad & \alpha_{231} = U_{231} \\
\alpha_{132} = U_{132} & & \alpha_{312} = U_{312} \\
\alpha_{213} = U_{213} & & \alpha_{321} = U_{321}
\end{array}$$

From this, one computes

$$\begin{array}{lll}
\alpha_{12} & = & \text{smaller} \left\{ U_{12},\ E_{\xi 12} \left[\alpha_{123} \right] \right\} \\
\alpha_{13} & = & \text{smaller} \left\{ U_{13},\ E_{\xi 13} \left[\alpha_{132} \right] \right\} \\
\alpha_{21} & = & \text{smaller} \left\{ U_{21},\ E_{\xi 21} \left[\alpha_{213} \right] \right\} \\
\alpha_{23} & = & \text{smaller} \left\{ U_{23},\ E_{\xi 23} \left[\alpha_{231} \right] \right\} \\
\alpha_{31} & = & \text{smaller} \left\{ U_{31},\ E_{\xi 31} \left[\alpha_{312} \right] \right\} \\
\alpha_{32} & = & \text{smaller} \left\{ U_{32},\ E_{\xi 32} \left[\alpha_{321} \right] \right\}
\end{array}$$

Then one computes

$$\begin{array}{lll}
\alpha_{1} & = & \text{smaller} \left\{ U_{1},\ E_{\xi 1} \left[\alpha_{12} \right],\ E_{\xi 1} \left[\alpha_{13} \right] \right\} \\
\alpha_{2} & = & \text{smaller} \left\{ U_{2},\ E_{\xi 2} \left[\alpha_{21} \right],\ E_{\xi 2} \left[\alpha_{23} \right] \right\} \\
\alpha_{3} & = & \text{smaller} \left\{ U_{3},\ E_{\xi 3} \left[\alpha_{31} \right],\ E_{\xi 3} \left[\alpha_{32} \right] \right\}
\end{array}$$

Finally, one establishes

$$\alpha_{\emptyset} = \text{smaller} \left\{ U_{\emptyset},\ E_{\xi} \left[\alpha_{1} \right],\ E_{\xi} \left[\alpha_{2} \right],\ E_{\xi} \left[\alpha_{3} \right] \right\}$$

One can note that α_{ijk} is the expected loss associated with performing experiments i, j, k -- in that order. The expression α_{ij} is the smaller of the expected losses associated with stopping or with continuing. Hence, it is the minimum expected loss (corresponding to the minimum expected loss procedure). This argument is repeated for α_i and for $\alpha_{\emptyset}$.

By this argument, one sees that α_{ϕ} is the minimum expected risk prior to experimenting.

To utilize this procedure, one should decide without experimentation if

$$\alpha_{\phi} = U_{\phi}$$

If this is not so, and

$$\alpha_{\phi} = E_{\mathfrak{z}}\left[\alpha_i\right],$$

then one should perform experiment e_i.
Having done this, one obtains the result $x_i = x_i^o$, and inquires if

$$\alpha_i(x_i^o) = U_i(x_i^o)$$

wherein one should decide without further experimentation. If this is not so, and

$$\alpha_i(x_i^o) = E_{\mathfrak{z}i}\left[\alpha_{ij}\right],$$

then the procedure calls for continuation by performing e_j, and etc. This is illustrated by the tree shown in figure C-2.

A much more formal proof can be argued on the basis of showing that the expected loss for any other partitioning of the outcome space into actions or experiments will be higher than that described above. This has been accomplished, but owing to its lack of heuristic appeal is not included in this report.

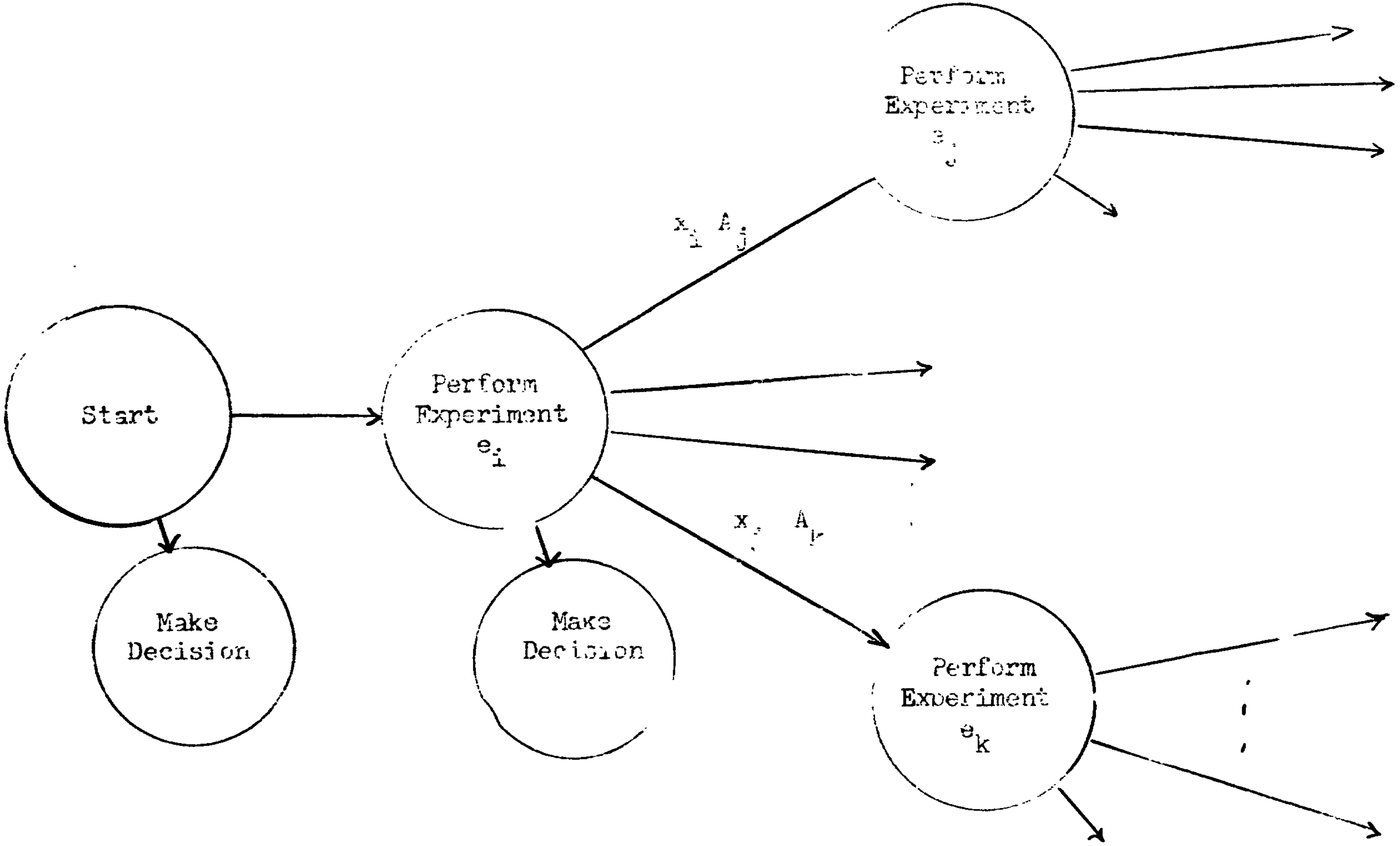


Figure C-2 Decision Tree, Path Depending on Experiment Outcome

Bibliography

1. Cooper, P. W., "The Hyperplane in Pattern Recognition", Melpar Technical Note 61/6, Melpar Applied Science Division, 1961.

2. Cooper, P. W., "The Hypersphere in Pattern Recognition, Melpar Technical Note 62/1, Melpar Applied Science Division, 1962.

3. Sebestyen, G. S., *Decision Making Process in Pattern Recognition*, MacMillan Company, New York, 1962.

4. Cooper, P. W., "The Hyperplane in Pattern Recognition", Cybernetics 5, 215-238.

5. Cooper, P. W., Hyperplanes, Hyperspheres, and Hyperquadrics as Decision Boundaries, *ONR-CCINS Symposium at Northwestern University*, appearing in Computers and Information Sciences, Spartan Books, Washington, D.C., 1963.

6. Sommerville, D. M. Y., *An Introduction to the Geometry of N-Dimensions*, Dover Publications, New York, 1958.

7. Cooper, P. W. and Cooper, D. B., Nonsupervised Adaptive Signal Detection and Pattern Recognition, *Information and Control*, 7, 416-444.

8. Blackwell, D. and Girshick, M. A., *Theory of Games and Statistical Decisions*, John Wiley, 1954.

9. Luce, R. D. and Raiffa, H., *Games and Decision, Introduction and Critical Survey*, John Wiley, 1957.

10. Thomas, J. B. and Wolfe, J. K., On the Statistical Detection Problem for Multiple Signals, *IEEE Trans on Information Theory*, Vol IT-8, pp 274-280, July 1962.

11. Ogg, F. C., Jr., A Note on Bayes Detection of Signals, *IEEE Trans on Information Theory*, Vol IT-10, pp 57-60, January 1964.

12. Wald, A, *Sequential Analysis*, Wiley and Sons, London, 1947.

13. Lee, R. C. K., *Optimal Estimation and Control*, MIT Press, 1964.

14. Beckenbach, E. F., *Modern Mathematics for the Engineer*, McGraw-Hill Book Co., 1956.

15. Kemeny, J. G., and Snell, J. L., Finite Markov Chains, D. Van Nostrand Co., Inc., 1960.

16. Jackson, A. S., Analog Computation, McGraw-Hill Book Co., Inc., 1960.

17. Tou, J. T., Modern Control Theory, McGraw-Hill Book Co., Inc., 1964.

18. Freeman, H., Discrete Time Systems, John Wiley and Sons, 1965.

19. Howard, R. A., Dynamic Programming and Markov Process, MIT Press, 1960.

20. Kunz, K. S., Numerical Analysis, McGraw-Hill Book Co., Inc., 1957.